



# **CALCULATOARELE ELECTRONICE DIN GENERAȚIA A CINCEA**

---

EDITURA ACADEMIEI  
REPUBLICII SOCIALISTE ROMÂNIA



ACADEMIA REPUBLICII SOCIALISTE ROMÂNIA

COMISIA „REVOLUȚIA ȘTIINȚIFICĂ ȘI TEHNICĂ”

INSTITUTUL DE CERCETARE ȘTIINȚIFICĂ ȘI INGINERIE TEHNOLOGICĂ  
PENTRU TEHNICĂ DE CALCUL ȘI INFORMATICĂ

# CALCULATOARELE ELECTRONICE DIN GENERAȚIA A CINCEA

EDITURA ACADEMIEI REPUBLICII SOCIALISTE ROMÂNIA

București, 1985



## P R E F A Ț Ă

În acest sfârșit de secol, microelectronica, informatica și ingineria genetică aduc cele mai mari schimbări tehnologice, nu numai în domeniile comunicațiilor, informației, automatizării și biotehnologiilor, ci în toate ramurile industriei. Ele produc noi revoluții tehnologice, iar primele două, microelectronica și informatica, însoțite de robotică și inteligență artificială ne îndreaptă cu pași rapizi către o nouă revoluție industrială.

Cum să nu fie atunci fascinantă străduința omului și a societății pentru cunoașterea din ce în ce mai aprofundată a proprietăților informației, pentru crearea de creiere electrono-informatică, pentru utilizarea acestora din urmă în mașini, roboți, în stații de lucru pentru oameni de știință, ingineri, economiști și oameni de artă și chiar ca noduri ale inteligenței sociale?

În contextul celor de mai înainte, apariția calculatorului electronic din generația a cincea va constitui un eveniment tehnologic deosebit cu implicații profunde asupra vieții economice, culturale și politice a societății.

Acest calculator va reprezenta o primă mare schimbare în însăși concepția și tehnologia calculatoarelor electronice. Dacă primele patru generații au fost determinate, în principal de electronică (tuburi electronice, tranzistori, circuite integrate), generația a cincea se bazează pe utilizarea conceptelor informaționale ale inteligenței artificiale. Cu alte cuvinte, stratul informațional inteligent este cel care structurează noul calculator, dar un strat informațional care va fi capabil să converseze cu omul în limbaj natural, prin text sau prin vorbire și, de asemenea, prin grafice, imagini sau alte mijloace.

Din punctul de vedere al utilizării, calculatorul electronic cu inteligență artificială va fi nu numai un asistent, ci, în primul rând un colaborator al omului în activitatea de cunoaștere, de sistematizare și utilizare a cunoștințelor, sub forma unor sisteme expert care acumulează cunoașterea naturală și artificială. Sistemele expert vor achiziționa și singure cunoștințe dacă li se anexează instrumentele necesare de interogare a naturii, transformate în echipamente periferice sau terminale informatice cu caracter specific.

Asemenea sisteme vor multiplica cei mai buni experți umani făcând disponibile cunoștințele și modul lor de a rezolva probleme pentru toți specialiștii dintr-un domeniu tehnologic, științific, medical, educațional etc. Noile calculatoare sub forma de sistem expert vor fi manuale informatizate de studiu, surse de informație și cunoștințe, vor da răspunsuri la problemele care li se pun, vor discuta cu specialiștii etc.

Calculatorul din generația a cincea va duce la o formă de umanizare nu numai a calculatoarelor electronice, ci și a tehnologiei în întregime



deoarece programarea și schimbul de informații cu noul mediu tehnic inteligent se va face în modurile familiare și naturale pentru om. Calculatoarele și noile sisteme tehnologice vor fi astfel, sperăm, mai prietenoase față de om.

În acest volum, specialiștii români din domeniile electronicii, tehnicii de calcul și informaticii trec în revistă conceptul calculatorului din generația a cincea și expun punctele lor de vedere. Este remarcabil faptul că în țara noastră se desfășoară activități pe linia de gândire a noii generații de calculatoare; mai mult, s-a realizat un prototip de calculator specializat pentru limbajul LISP necesar inteligenței artificiale. În același timp, lucrurile sînt examinate cu realism în contextul industriei și economiei românești.

Dezvoltarea microelectronicii și informaticii în țara noastră, începută acum 20 de ani ca urmare a hotărîrilor Congresului al IX-lea al Partidului Comunist Român, consecință a eforturilor de modernizare a României, inițiate de tovarășul Nicolae Ceaușescu, Secretarul general al Partidului, a asigurat o anumită maturizare a acestor domenii, atât în cercetarea științifică și învățămînt, cît și în industrie. De aceea, ținînd cont și de directivele Congresului al XIII-lea din anul 1984 care prevăd o masivă creștere a electronicii și aplicațiilor ei, se impune o luare de poziție și a specialiștilor români față de conceptul calculatorului electronic din generația a cincea.

Calculatorul de generația a cincea, un proiect în curs de realizare, a trezit un mare interes, multe ambiții și a determinat programe speciale de cercetare și inginerie tehnologică în multe țări. Volumul de față reunește atitudinea noastră de azi în raport cu acest proiect.

Ce vom gândi și vom scrie mîine?

Prof. MIHAI DRĂGĂNESCU,  
membru corespondent al Academiei  
Republicii Socialiste România



## CUPRINS

MIHAI DRĂGĂNESCU, Calculatorul din generația a cincea : un eveniment tehnologic, cultural și politic	9
ADRIAN DAVIDOVIGIU, Prezentarea proiectului japonez pentru calculatorul de generația a cincea	18
EMIL TUDOR, LUCIAN NICA, MIHAI MÂRȘANU, Concepte privind arhitectura sistemelor de calcul din generația a cincea	47
VICTOR MEGHEȘAN, NICOLAE COSTAKE, Generația a cincea de calculatoare electronice și industria de tehnică de calcul românească	60
VICTOR MEGHEȘAN, NICOLAE COSTAKE, MIHAI MÂRȘANU, Direcțiile de dezvoltare ale tehnicii de calcul românești la orizontul 1990	75
GORUN MANOLESCU, Calculatoarele de generația a cincea și unele probleme ale cercetării-proiectării asistate de calculator	88
ADRIAN PETRESCU, Sisteme sistolice de prelucrare a datelor	101
GHEORGHE M. ȘTEFAN, AUREL PĂUN, Compatibilizarea funcție-structură ca mecanism al evoluției arhitecturale	113
RADU M. BÂRSAN, Tehnologii pentru circuite integrate pe scară foarte mare	136
CRISTIAN GIUMALE, Generația a cincea de calculatoare — un pas spre programarea naturală?	146
AUREL PĂUN, GHEORGHE M. ȘTEFAN, ANDY BIRNBAUM, VIRGIL BISTRICEANU, DIALISP — experiment de structurare neconvențională a unei mașini LISP	160
POSTFAȚĂ	177
CONTENTS	181
СОДЕРЖАНИЕ	182



# CALCULATORUL DIN GENERAȚIA A CINCEA : UN EVENIMENT TEHNOLOGIC, CULTURAL ȘI POLITIC

MIHAI DRĂGĂNESCU \*)

FIFTH-GENERATION COMPUTER: A TECHNOLOGICAL, CULTURAL AND POLITICAL EVENT. The fifth generation computer is considered as a technological, cultural and political event. This computer will amplify the wave of the second industrial revolution and will have a decisive impact on the information-oriented society. The requirements and the functions of the fifth generation computer, as seen by the Japanese specialists, are considered; the world reaction to the Japanese project is presented; some possible human consequences are envisaged.

## 1. Introducere

Lansarea proiectului calculatorului de generația a cincea de către Japonia [1] în anul 1981 a fost rezultatul unei gândiri tehnologice avansate îmbinată cu o viziune asupra viitorului sub forma „societății informaționale” [2].

Din punctul de vedere al logicii interne a tehnologiei, după trecerea de la prima generație de calculatoare, cu tuburi electronice, la generația a doua cu tranzistori, dezvoltarea circuitelor integrate a impus două generații de calculatoare electronice: a treia cu circuite integrate și a patra cu circuite integrate pe scară largă și foarte largă (vezi tabelul 1). Dacă primele patru generații au apărut ca urmare a unor salturi tehnologice în domeniul dispozitivelor și circuitelor electronice, generația a cincea nu mai este impulsionată, în principal, de o tehnologie hardware, ci de una software: aceea a inteligenței artificiale. Totuși, se impun noi progrese, se spune chiar noi străpungeri în domeniile tehnologiilor hardware și software pentru a se realiza un asemenea calculator. Fezabilitatea proiectului japonez a fost rapid recunoscută de către specialiștii din toate țările dezvoltate, cu gândul că și în cazul în care numai o parte din obiectivele propuse vor fi atinse, se vor realiza progrese uriașe în domeniul calculatoarelor electronice.

Dincolo de calculatorul din generația a cincea se poate prevedea apariția calculatoarelor moleculare, utilizând structuri de enzime și proteine [3] sau polimeri organici cu proprietăți speciale. Dacă ținem seama de faptul că noua revoluție industrială se bazează pe microelectronică, inteligență artificială și automatizare flexibilă, intensitatea maximă a acestui proces revoluționar va urma apariției calculatorului din generația a cincea. În același timp se pregătește unda unei revoluții biologice și biochimice la care ar putea corespunde peste 25 ÷ 35 de ani și un calculator molecular. Singurul lucru care ar mai rămâne după aceea ar fi apari-

\*) Institutul politehnic din București.



Tabelul 1

Generații de calculatoare electronice\*)

Gene- rația	Perioada	Principala tehnologie hardware	Alte tehnologii hardware	Limbaje	Alte proprietăți
I	1946—1956	Tuburi electronice	Tambur mag- netic	de asamblare	MI (memorie internă) = 2 ki- loocteți. V (vi- teză) = 10 KIPS (kilo- instrucțiuni pe secundă)
II	1957—1963	Tranzistori	Memorii din ferite	de nivel înalt : COBOL, FORTRAN etc.	MI = 32 kiocteți V = 200 KIPS
III	1964—1981	Circuite integrate	Memorii semi- conductoare Discuri mag- netice	de nivel foarte înalt : PASCAL LISP, Limba- je grafice etc.	MI = 2 M octe- teți. V = 5 MIPS (mega- instrucțiuni pe secundă)
IV	1982—1989	Circuite integrate pe scară largă și foarte largă	Memorii cu bule magnetice Discuri optice	ADA limbaje orientate obiect	MI = 8 M oc- teți V = 30 MIPS Supercalcula- toare
V	din 1990	Circuite integrate pe scară extrem de largă (inclusiv tridimensionale)	Arhitecturi paralel	Limbaje con- curente LIM- BAJ NATU- RAL Programare funcțională	INTELIGEN- ȚA ARTIFI- CIALĂ. V = de la 1 GIPS (giga- instrucțiuni pe secundă) la 1000 GIPS Vedere artifi- cială Tehnologia vorbirii

\*) după Robert E. Kahn, I.E.E.E. Spectrum, 1983, november, p. 37.

ția unui calculator viu, cu o inteligență artificială vie (IAV), ceea ce ar implica și atașarea unor structuri de tip ADN sau ARN la un calculator molecular.

Dar să nu ne gândim atât de departe, rămânând în cadrul temei date, remarcând că odată cu calculatoarele din generația a cincea se produce o schimbare radicală în domeniul computerelor, care nu vor mai fi propriu-zis calculatoare, ci, cum le-au denumit specialiștii japonezi, sisteme de procesare a informației de cunoaștere (Knowledge Information Processing Systems = KIPS), adică sisteme create pentru îndeplinirea funcțiilor de inteligență artificială. Importanța unor asemenea sisteme este astfel subliniată de Edward A. Feigenbaum, unul din principalii creatori ai sistemului expert DENDRAL, și Pamela Mo. Corduck : „Cea mai mare parte a muncii în lume este nematematică, prin natura ei. Numai un mic segment de activitate are ca nucleu tipurile de formule pe care le întâlnim



în fizica aplicată și inginerie. Chiar și în asemenea științe „puternice” cum este chimia, cea mai mare parte din gândire se realizează prin inferență, nu prin calcule. Același lucru este valabil pentru biologie, cea mai mare parte a medicinei și tot ceea ce se referă la drept. Aproape întreaga gândire a conducerii afacerilor se face prin inferență simbolică, nu prin calcule. Pe scurt, aproape întreaga gândire a oamenilor de profesie se desfășoară prin raționamente și nu prin calcule” [4].

Dacă primele patru generații de calculatoare electronice au fost construite pentru calcule, chiar dacă au fost utilizate treptat și pentru prelucrări simbolice, generația a cincea va fi un procesor de informație aproape sub orice formă utilă omului (limbaj natural, voce, imagine), fără a se ridica însă la nivelul unui procesor mental deplin, din momentul în care nu putem pune semnul egalității între inteligența artificială și inteligența naturală (umană) [5].

Apariția calculatorului din generația a cincea este prevăzută în anii 1990. Între timp, datorită automatizării flexibile și extinderii roboților, dar și a sistemelor expert utilizând calculatoare clasice, calculatoare LISP (un asemenea prototip este realizat și în țara noastră, iar despre el se va vorbi în această sesiune [6]), eventual calculatoare PROLOG, procesul celei de-a doua revoluții industriale se va fi instalat deplin, dar maximum lui va fi atins odată cu apariția și utilizarea calculatoarelor din generația a cincea. Această generație de calculatoare va definitiva orientarea informațională a societății. Aceasta va fi o mare schimbare în societatea umană care va utiliza forțe de producție radical modificate, cu toate implicațiile care decurg de aici.

## 2. Cerințe și funcțiuni

Am încercat în alte lucrări să privim modul în care tehnologia s-ar putea racorda la cerințele vieții spirituale ale omului și ale unei civilizații socioumane, denumind *tehnologie politică* un asemenea mod de abordare. Comunismul, care în lumina noilor desfășurări tehnologice va fi o societate orientată informațional, va trebui să îmbine tehnologia nouă cu cerințele multilaterale ale omului și societății. Iar această tehnologie nouă va avea în inima ei procesorul electronic de generația a cincea. Ea va impune ca munca umană să fie redefinită: ce anume trebuie să muncească omul și ce anume va lăsa în seama sistemelor automatizate și informatizate? Cum anume își va desfășura omul viața într-un mediu informațional tehnic și inteligent? Care va fi sensul vieții lui?

Tehnologia politică va trebui, de fapt, să abordeze invers problemele: de la ce ideal de societate, de viață pentru om, pornim pentru a defini funcțiunile pe care le cerem din partea tehnologiei? Tehnologia informațională conferă tehnologiei, în ansamblul ei, o mare plasticitate pentru a realiza o funcțiune sau alta în societate sau în raport cu omul. În acest sens, întreaga tehnologie va deveni funcțională. Electronica a devenit funcțională odată cu apariția microprocesorului [7] deoarece cu ajutorul acestei componente microelectronice se pot programa și realiza o mare varietate de funcțiuni. Calculatorul din generația a cincea este el însuși funcțional. Inițiatorii proiectului acestui procesor au definit astfel funcțiunile care i se cer [8]:

— funcțiunea inteligenței, similară inteligenței omului, utilizabilă în raport cu unul sau cît mai multe domenii de cunoaștere;



- funcțiunii similare simțurilor omului : auz, văz, simț tactil etc., dar și vorbire, exprimare grafică sau în imagini ;
- funcțiunea de conversație în limbaj natural cu omul ;
- funcțiunea de a acționa în locul omului într-o serie de activități ;
- funcțiunea de a achiziționa cunoaștere și de a colabora cu omul în explorarea unor domenii noi ;
- funcțiunea de consultant pentru om ;
- funcțiunea de a se adapta și configura la diferite tipuri de activități ;
- funcțiunea de a scrie (autoscrie) programe informatice pornind de la cerințe specificate ;
- funcțiunea de a se autorepara etc.

Interesant este faptul că primul raport general al specialiștilor japonezi referitor la calculatorul din generația a cincea începe cu *cerințele sociale față de calculatoare* în anii 1990. În prima frază a acestui raport se arată : „În anii 1990 când se așteaptă ca sistemele de calcul din generația a cincea să fie larg utilizate, sistemele de procesare a informației vor fi uneltele centrale în toate domeniile activității sociale, care includ economia, industria, arta, știința, administrația, relațiile internaționale, educația, cultura, viața zilnică și așa mai departe” [9].

Specialiștii japonezi și-au pus două întrebări fundamentale, spunem noi, de tehnologie politică [10] :

- a) Ce tipuri de nevoi sociale vor putea satisface noile tehnologii ?
- b) Ce impact, ce consecințe vor avea noile calculatoare când ele vor fi instalate ?

Aceste întrebări nu sînt independente deoarece noi nevoi apar sau pot fi imaginate ca urmare a impactului acestor calculatoare, sau după cum spune Hajime Karatsu, „viitorul nu este extinderea trecutului, ci ceva care trebuie creat din nou” [11].

Care sînt cerințele sociale prevăzute de japonezi ?

În primul rînd, creșterea productivității muncii în domeniile care au astăzi o productivitate redusă: în birouri, proiectare, agricultură, medicină, educație, servicii, administrație guvernamentală. Este exact procesul extinderii celei de a doua revoluții industriale la alte domenii decît industria propriu-zisă, industrializîndu-le și pe acestea, desigur sub forma unei industrializări noi, informatice.

Tehnologia actuală, cu unele perfecționări, este suficientă pentru automatizarea flexibilă a industriei de azi. Într-adevăr, specialiștii japonezi și americani consideră că în decurs de 10 ani automatizarea flexibilă, numită în ultimul timp și automatizare condusă de informație (data-driven automation [12]) va revoluționa actuala industrie. Fără a ridica productivitatea muncii și în domenii ca cele menționate mai înainte (birouri, proiectare, programare etc.) s-ar ajunge la noi distorsiuni sociale, astfel încît calculatorului din generația a cincea și de fapt dezvoltarea inteligenței artificiale apare ca o necesitate obiectivă, de echilibrare a consecințelor sociale ale automatizării flexibile.

În al doilea rînd, spun specialiștii japonezi, a face față competiției internaționale și a dezvolta cooperarea mondială. Nu este nevoie să mai insistăm asupra intenției de a se prelua conducerea tehnologică și economică în lume.

În al treilea rînd, economisirea resurselor naturale și economisirea de energie.



În al patrulea rând, a face față unei societăți cu o populație cu vîrstă mărită datorită creșterii vieții medii a omului, față de care populație trebuie să se asigure suportarea cheltuielilor sociale, a unui sistem medical informatizat perfecționat și a unui sistem educațional informatizat îndreptat și înspre populația în vîrstă.

În al cincilea rând, de a nu lăsa ca noile calculatoare să se întoarcă împotriva omenirii[13], deoarece calculatoarele vor fi unelte cu inteligență care vor coexista cu ființele umane. Pot fi reamintite aici cele trei principii enunțate de Asimov cu privire la roboți și cele două principii pe care le-am enunțat cu privire la inteligența artificială (tabelul 2), principii care ar trebui programate.

Desigur ne putem imagina și alte cerințe și consecințe implicate de procesoarele cu inteligență artificială, deschizîndu-se din acest punct de vedere un cîmp de posibilități. Cea mai importantă regulă ar putea fi aceea ca inteligența artificială devenită inteligență socială să influențeze omul și societatea spre bine, frumos și civilizație.

Tabelul 2

Reguli de respectat de către roboți și inteligența artificială.

Regulile lui Asimov privitoare la roboți	<ol style="list-style-type: none"> <li>1. Un robot nu poate periclita viața unui om sau, prin inactivitatea sa, să lase ca această viață să fie periclitată.</li> <li>2. Un robot trebuie să asculte de ordinele primite din partea unei ființe umane, cu condiția ca să nu contrazică prima regulă.</li> <li>3. Un robot trebuie să-și apere existența proprie, cu condiția de a nu contraveni regulii 1 sau 2.</li> </ol>
Reguli privind inteligența artificială	<ol style="list-style-type: none"> <li>1. Un sistem de inteligență artificială nu va accepta sau formula o soluție unei probleme care ar putea periclita existența speciei umane, astăzi sau în viitor, sau ar putea contrazice principiile civilizației.</li> <li>2. Un sistem de inteligență artificială nu are drepturi politice, nu poate vota și nu poate lua decizii fără o supraveghere umană.</li> </ol>

### 3. Reacția mondială față de proiectul japonez

Proiectul japonez, elaborat sub finanțare guvernamentală, a fost urmat rapid de înființarea unui institut special pentru elaborarea calculatorului din generația a cincea, ICOT (Institute for New Generation Computer Technology), condus de Kazuhiro Fuchi, ca unitate independentă însă finanțată de opt mari companii: Fujitsu, Hitachi, Matsushita Electric, Mitsubishi Electric, Electric, NEC ș.a. S-a trecut la publicarea unei reviste dedicate noului calculator, „New Generation Computer”, editată de ICOT.

Între 6—9 noiembrie 1984 a avut loc la Tokio o conferință internațională privind sistemele de calcul din generația a cincea (tabelul 3).

Reacția cea mai puternică la proiectul japonez s-a manifestat în S.U.A., apoi în Marea Britanie și în restul Europei. În S.U.A. apar cărți și numere întregi ale unor publicații periodice de specialitate [16] despre proiectul acestui calculator. Autori americani remarcă cu oarecare nemulțumire față de modul de inovare la scară socială în domeniul tehnologiilor de vîrf din țara lor că „programul japonez este privit de mulți ca fiind bazat, în mod esențial, pe rezultatele cercetării din S.U.A. din ultimii



Tabelul 3

Elemente privind tematica conferinței internaționale asupra calculatorului din generația a cincea (Tokio, 1984).

Fundamentele programării logice	Semantică și pragmatică formală Modele computaționale Aspecte psihologice și filosofice
Limbaje pentru programare logică	Limbaje de programare orientate spre obiect și cu funcționare în paralel Meta-nivel de inferență și control Mediu inteligent de programare Sinteza programelor Verificarea programelor
Arhitectura noii generații de calculatoare	Mașini de inferență Mașini baze de cunoștințe Arhitecturi de procesoare în paralel Arhitecturi VLSI Noi interfețe om-mașină
Aplicații ale noii generații de calculatoare	Achiziția și reprezentarea cunoașterii Sisteme expert Înțelegerea limbajului natural Grafică și vedere artificială
Impactul noii generații de calculatoare	Cultural și social Economic Industrial Educațional Internațional

20 de ani, incluzând domeniul inteligenței artificiale” [17]. Dar răspunsul nu a întârziat prea mult deoarece după cum remarcă un specialist american „impactul potențial al unei asemenea noi generații, virtual asupra oricărui sector de activitate profesională și personală a impulsionat, cum puține tehnologii precedente au făcut-o, forurile decizionale din lume” [18]. Într-adevăr, în momentul de față, guvernul S.U.A., prin DARPA, finanțează cercetări în valoare de 1 miliard de dolari, în perioada 1983–1990, pentru realizarea noii generații de calculatoare. Se mai poate menționa că deja DARPA finanțează un alt program de 1 miliard de dolari pentru circuite integrate pe scară largă și ultra largă, de viteză din ce în ce mai mare. Pentru prima oară în S.U.A., un număr de 11 companii (între care Digital Equipment Corporation, Control Data, Honeywell, Sperry, Motorola, National Semiconductor ș.a.). au format împreună o corporație de cercetare (The Microelectronics and Computer Technology Corporation — MCC) finanțată în anul 1984 cu 50 milioane de dolari. Acest institut de cercetare, care este în mod evident un răspuns la ICOT-ul japonez, are programe în vederea realizării calculatorului din noua generație, cu o durată de pînă la zece ani. La cele de mai sus se adaugă consorțiul SRC (Semiconductor Research Cooperative) care cuprinde grupuri de cercetare (în total 150 de cercetători principali) din 30 de universități, consorțiu finanțat de 23 de companii din domeniul semiconductorilor, cu un buget total de 13 milioane dolari (în anul 1984 disponibil prin contracte, cu un aparat de coordonare format din 12 persoane). În fine, statul mai finanțează un grup de cinci universități din Carolina de Nord pentru cercetări în domeniul microelectronicii, cu un buget de 10 milioane de dolari anual,



pe timp de cinci ani (MCNC — Microelectronics Center of North Carolina, cu o participare de 15 % din partea industriei) [19]. La toate aceste organizații se mai adaugă propunerea de a se reuni cercetătorii din domeniile implicate în aceste acțiuni printr-o rețea de calculatoare, terminale, linii și noduri pentru transmisii de date, într-o comunitate în care timpul de comunicare al noilor rezultate se reduce dramatic. Un asemenea procedeu va avea cu siguranță un mare impact pentru cercetarea științifică în viitor.

După cum se poate observa, răspunsul american este masiv, declanșându-se o adevărată competiție mondială pentru noile tehnologii de vîrf din jurul calculatorului din generația a cincea. Acest obiectiv nu mai este unul japonez, ci a devenit global.

Există și un răspuns englez, Marea Britanie inițiind un program de 500 milioane de dolari pe 5 ani pentru cercetări în toate domeniile tehnologice necesare calculatoarelor din generația a cincea. Comunitatea Economică Europeană lansează programul ESPRIT (European Strategic Programme in Information Technology) în aceeași direcție, finanțat de guvernele participante și de 12 companii importante (ICL — Anglia, Bull — Franța, Siemens — R.F.G., Olivetti — Italia ș.a.), la o valoare propusă de peste 1,5 miliarde dolari pentru primii cinci ani. Comisia interguvernamentală pentru tehnica de calcul a țărilor socialiste a trecut la dezbaterea problemelor noii generații de calculatoare și desigur programe de cercetare au și început în unele țări socialiste.

A sosit momentul ca și specialiștii români să ia o atitudine publică față de proiectul acestui calculator. Cîmpul de activitate pentru acest calculator coincide de fapt cu aproape întreg domeniul tehnologiei informației: arhitecturile noi de calculatoare (non - von Neumann), microelectronica, limbaje de programare, ingineria programelor informatice, inteligența artificială, informatica limbajului natural, vederea artificială, tehnologia vorbirii, bănci de date și baze de cunoștințe, rețele de calculatoare etc. În fiecare din aceste domenii sînt posibile progrese și în țara noastră, chiar dacă nu ne putem încumeta singuri, astăzi, la un proiect de ansamblu.

În același timp trebuie să fim conștienți de importanța acestui proiect pentru întreaga tehnologie în viitorul apropiat și de fapt pentru istoria omenirii.

#### **4. Consecințele umane ale noului calculator**

Consecințele calculatorului inteligent asupra omului depind de domeniile de utilizare, de influența asupra muncii omului, asupra inteligenței lui, ca și asupra comportamentului său psihologic și imaginii sale despre sine. Putem astfel prevedea:

— Schimbarea întregului sistem de învățămînt și educație în viitor, din momentul în care acest calculator va fi o carte inteligentă cu care se conversează și care va permite fiecăruia să progreseze în ritmul său propriu; cele mai bune „cursuri” din lume vor putea fi imediat disponibile prin utilizarea calculatoarelor traducătoare dintr-o limbă în alta; învățămîntul la terminal (calculator propriu) conectat la o rețea de calculatoare inteligente specializate, deci la sisteme expert, la bănci de date, de cunoștințe va putea deveni o realitate. Acest învățămînt nu va elimina contactele umane directe, deși modificări vor interveni.

— Schimbarea caracterului muncii omului deoarece toate activitățile fizice grele, periculoase, de rutină, repetitive vor reveni roboților și



automatelor inteligente. Schimbările sînt valabile și pentru munca intelectuală, treburile administrative de rutină fiind preluate de calculatoare, de asemenea sistemele expert vor fi o prezență permanentă în munca de cercetare științifică și inginerie, în producție, încît principala activitate a omului va fi aceea de a crea și construi noi sisteme informatice și noi sisteme încorporînd calculatoare inteligente, cu alte cuvinte omul va trece la o activitate preponderent informațională și mai ales cu caracter de obținere de cunoaștere nouă.

— Omul se va îndrepta din ce în ce către o activitate creativă, de căutare a noului dincolo de ce a fost cuprins în sistemele tehnice inteligente. Dacă nu va găsi noul, atunci îl va inventa. Această căutare și putere de creație permanentă îi va asigura supraviețuirea întrucît sînt speranțe ca el să poată descoperi secrete profunde ale materiei.

— O mare parte din creativitatea și inventivitatea sa se va îndrepta către artă, sub toate formele, către îmbogățirea sa spirituală și moduri noi de a consuma timpul disponibil împreună cu semenii săi.

— Starea biologică a omului va putea fi din ce în ce mai bine controlată, în sensul pozitiv al cuvîntului, sistemele medicale vor deveni cu adevărat sisteme ale menținerii sănătății, calculatoarele expert avînd posibilitatea să urmărească în mod corelat parametrii care determină starea de sănătate a fiecăruia, oferind medicilor interniști informații pe care altfel le-ar fi putut asambla cu multă dificultate.

— Meseria nouă a viitorului determinată de noile calculatoare va fi aceea a *ingineriei cunoașterii*. Ea se referă la elaborarea metodelor de reprezentare a cunoașterii, de achiziție automată a cunoașterii și de realizare a sistemelor de cunoaștere, de la cele mai simple pînă la sisteme în rețea, în care conlucrează diverse sisteme expert.

Se poate prevedea că apariția noului calculator electronic ca formă de materializare a inteligenței artificiale va conduce la modificarea imaginii omului despre om. Impactul psihologic al noului calculator asupra omului va fi deosebit de puternic: omul va avea de ales între a se considera similar unui asemenea calculator sau în a găsi calități noi substanței sale biologice. Nu putem prezice exact reacția sa în fața unui mediu tehnologic inteligent, însă ceea ce se poate spune sigur este faptul că el va căuta să se delimiteze într-un fel de acest mediu. Va fi oare ajutat în acest efort de descoperirile care se vor face în domeniul biologiei și, în general, în știință? Dacă da, ceea ce sîntem aproape convinși, atunci el va descoperi adevărul despre sine, iar la această descoperire va fi contribuit și ingineria inteligenței artificiale.

Probabil calculatorul din generația a cincea va reprezenta vîrfurile tehnologiei bazate pe cunoștințele despre materia nevie. Ne așteaptă în viitor potențialitățile materiei vii și ale materiei profunde. Aventura cunoașterii, tehnologiei și spiritului uman are încă un drum lung de parcurs.

#### REFERINTE BIBLIOGRAFICE

1. T. MOTO-OKA (ed), *Fifth generation computer systems*, Proceedings of the International Conference on Fifth Generation Computer Systems, Tokyo, october 19—22, 1981, Amsterdam, North Holland, 1982.
2. YONEJI MASUDA, *The information society as post-industrial society*, Tokyo, Institute for the information society, 1980.



3. MARK A. FISCHETTI, *Key scientists say that molecular computing needs more funding*, The Institute (News Supplement to I.E.E.E. Spectrum), january, p. 1/22, 1984.
4. EDWARD A. FEIGENBAUM, PAMELA MC. CORDUCK, *The fifth generation, artificial intelligence and Japan's computer challenge to the world*, Reading, Massachusetts, Addison-Wesley, p. 18, 1983.
5. MIHAI DRĂGĂNESCU, *A doua revoluție industrială*, București, Edit. tehnică, 1980.
6. AUREL PĂUN, GHEORGHE ȘTEFAN, ANDY BIRNBAUM și VIRGIL BISTRICEANU, *DIALISP — experiment de structurare neconvențională a unei mașini LISP* (în acest volum, p. 160)
7. MIHAI DRĂGĂNESCU, *Viitorul electronicii și informaticii*, în *Viitorul electronicii și informaticii*, sub red. Mircea Malița și Mihai Drăgănescu, București, Edit. Academiei, p. 13 — 18, 1979.
8. T. MOTO-OKA et al., *Challenge for knowledge information processing systems ; Preliminary report on fifth generation computer systems*, în vol. cit. la ref. [1], p. 5—6 și 20—22.
9. Idem, p. 3.
10. HAJIME KARATSU, *What is required of the 5<sup>th</sup> generation computer — social needs and its impact*, în vol. cit. la ref. [1], p. 93.
11. Idem, p. 93.
12. \* \* \* I.E.E.E. Spectrum, 1983, may, special issue : *Data driven automation, toward a smarter enterprise*.
13. T. MOTO-OKA et al, op. cit., p. 15.
14. MIHAI DRĂGĂNESCU, *Social intelligence*, în *Inteligența artificială și robotică*, sub red. Mihai Drăgănescu, București, Edit. Academiei, 1983, p. 17.
15. vezi, spre exemplu, ref. [4].
16. spre exemplu, I.E.E.E. Spectrum, 1983, november, număr dedicat noii generații de calculatoare.
17. EDWARD A. TORRERO, *Tomorrow's computer*, I.E.E.E.Spectrum, november, p. 35, (1983).
18. Idem.
19. după MARK A. FISCHETTI, p. 51—53, 55—56, 59—63, Robert S. Cooper and Robert E. Kahn, p. 53—55, Erich Bloch, p. 56—57, James D. Meindl, p. 57—58 și Richard B. Fair, p. 59, 69, în I.E.E.E. Spectrum, november (1983).



# PREZENTAREA PROIECTULUI JAPONEZ PENTRU CALCULATORUL DE GENERAȚIA A CINCEA

ADRIAN DAVIDOVICIU \*)

AN OVERVIEW ON THE JAPANESE PROJECT FOR DEVELOPMENT OF THE FIFTH-GENERATION COMPUTER. Japan, followed also by some other countries, is engaged in a very important, large scale, long range, scientific and technological project. The objective is to develop a new generation computer system with reasoning capabilities, ability of processing large amount of knowledges and to natural language communication with the human users. The paper is an overview on the nature, significance and main topics (objectives) of Japan's Fifth Generation Computer Systems Project, aimed at the accomplishment of basic research for important innovations in future computer technology. The paper is based on recent reports and papers of worldwide specialists who have direct contacts with the Japanese project.

## 1. Introducere

Ca introducere aș dori să citez doi specialiști americani — Edward A. Feigenbaum și P. McCorduck [1] — care au vizitat instituțiile din Japonia implicate în cercetările legate de generația a cincea de calculatoare și care au reușit să sintetizeze foarte plastic ceea ce și-au propus japonezii, prin următoarele :

„Japonezii au întrezărit aur pe niște culmi mai îndepărtate și au început să acționeze în acea direcție. Strategii din Japonia văd în industria de calculatoare ceva vital pentru viitorul economic al țării lor și și-au propus, ca un scop național, de a deveni „numărul unu” în acest domeniu, în ultima jumătate a deceniului anilor 1990. Ei și-au propus nu numai să domine formele tradiționale ale industriei de calculatoare dar și de a realiza o „industrie a cunoașterii” („knowledge industry”), în care cunoașterea în sine să fie un produs vandabil cum sînt astăzi petrolul sau alimentele. Cunoașterea va deveni o nouă bogăție națională”.

Desigur că pentru a realiza un asemenea obiectiv japonezii și-au propus atît o strategie cît și o tactică adecvată.

*Strategia* japoneză este de a evita o confruntare directă pe piața calculatoarelor, dominată actualmente în continuare de firme americane; japonezii au preferat să scruteze domenii cu mare potențial economic la nivelul anilor '90, domenii pentru care eventual firmele americane, mai pragmatice în abordarea pe termen mai scurt a pieții, să fie deci mai vulnerabile, mai ușor de concurat.

*Tactica* japoneză în acest domeniu este de a organiza un Program național de cercetare și dezvoltare tehnologică în domeniul sistemelor de

---

\*) Institutul de cercetare științifică și inginerie tehnologică pentru tehnică de calcul și informatică.



prelucrare electronică a informației bazată pe cunoaștere sau de prelucrare a cunoștințelor (Knowledge Information Processing Systems, s-au prescurtat KIPS) cunoscut și sub denumirea generică de Generația a cincea de calculatoare.

## 2. Esența ideii proiectului japonez

Desigur că pentru a înțelege importanța proiectului japonez este important a înțelege ce anume înțeleg japonezii prin sisteme de prelucrare electronică a cunoștințelor, spre deosebire de sistemele de prelucrare a datelor sau a informației. Va trebui totodată să analizăm circumstanțele care au determinat și au favorizat apariția în Japonia a unui asemenea proiect, proiect care de fapt își propune să valorifice la scară industrială *tehnologia cunoașterii* a cărei rădăcini principale sînt totuși în cercetările științifice din S.U.A. și Anglia.

Semnificația atribuită de japonezi termenului de *prelucrare a cunoștințelor* marchează o schimbare de concepție în sensul trecerii de la prelucrarea actuală relativ brută, primară a datelor (bazată pe o prelucrare aritmetico-logică a datelor) la o prelucrare inteligentă a cunoștințelor, bazată pe funcții de inteligență artificială, de prelucrare simbolică a informației, inclusiv inferență logică.

După cum se știe, în marea ei majoritate lumea înconjurătoare nu este matematică prin natura ei. Doar segmente limitate dintr-un domeniu de activitate pot opera exclusiv cu formule sau alte expresii sau funcții matematice, în special în tehnică, fizică ș.a. Chiar și în domenii cu o mare încărcătură științifică cum ar fi chimia, biologia, medicina etc. majoritatea raționamentelor se realizează prin *inferențe* și nu prin calcule. Aproape în întregime deciziile în sistemele de conducere se fac pe bază de inferențe simbolice și nu pe bază de calcule matematice. Cu alte cuvinte, în cea mai mare parte a activității lor, profesioniștii (specialiștii) din diferitele domenii efectuează raționamente și mai puțin calcule. Deci, este cît se poate de normal ca aceștia să dorească să dispună de mijloace care să-i asiste în activitate, care să permită o automatizare a raționamentului, a inferențelor simbolice.

Asemenea metode de raționament și prelucrări simbolice se utilizează de cîțiva ani (din ce în ce mai mult în ultimii 8-10 ani) folosind echipamentele de calcul actuale de generațiile a patra, trei și jumătate și chiar trei, elaborîndu-se programe adecvate pentru asemenea tipuri de prelucrări, dezvoltîndu-se chiar și instrumente de programare adecvate (cum ar fi limbajele din familia LISP, limbajul PROLOG ș.a.) și chiar unele echipamente mai specializate (de exemplu, mașină LISP).

Majoritatea aplicațiilor actuale care prezintă asemenea caracteristici de prelucrare de cunoștințe, de inferențe și manipulare simbolică sînt din clasa de aplicații denumite *sisteme expert*, încercînd să reproducă modul de a raționa, de exemplu al unui specialist-medic atunci cînd stabilește un diagnostic sau al unui specialist-geolog atunci cînd stabilește perimetrul de investigație geologică în scopul descoperirii unor anumite substanțe minerale.

Totuși, sistemele expert actuale, bazate pe actualele echipamente de calcul, au performanțe limitate în ceea ce privește puterea și viteza de lucru, fiind încă primitive în raport cu performanțele umane. Manipularea (prelucrarea) de cunoștințe la scară mare, cît de cît apropiată de inteligența



umană, necesită mijloace hardware (echipamente de calcul) și software (programe) care depășesc cu câteva ordine de mărime generația actuală de calculatoare, iar proiectul japonez pentru generația a cincea de calculatoare își propune, pentru prima dată în istoria calculatoarelor și a societății umane un asemenea obiectiv.

Pe bună dreptate, generația a cincea de calculatoare, așa cum a fost definită de japonezi, nu va reprezenta doar o nouă evoluție tehnologică așa cum s-au petrecut lucrurile cu primele patru generații de calculatoare ci în primul rând o revoluție, o nouă eră a calculatoarelor, marcată, după cum s-a arătat, prin trecerea de la prelucrarea datelor la prelucrarea cunoștințelor.

De remarcat că japonezii, în paralel cu lansarea proiectului național pentru generația a cincea de calculatoare nu neglijează cercetările științifice și dezvoltarea tehnologică pentru perfecționarea calculatoarelor convenționale. În Japonia există și un proiect național de realizare a unui calculator super rapid (National Super Speed Computer Project—NSSCP). La acest program, care urmărește realizarea în jurul anului 1989 a unui calculator de circa 1 000 ori mai rapid decât cele mai rapide calculatoare de la începutul deceniului anilor '80, participă șase din cele mai importante firme industriale producătoare de calculatoare: Fujitsu, Hitachi, Nippon Electric Corp., Mitsubishi, Oki și Toshiba, sub conducerea Laboratorului Național Electrotehnic Japonez. Se semnalează, de asemenea, o serie de alte proiecte în domeniile de vîrf ale calculatoarelor, finanțate de guvernul japonez sau de firme constructoare, cum ar fi cele în domeniul microelectronicii, prelucrarea imaginilor, înțelegerea limbajului natural ș.a. Un grup de cercetători americani de la Livermore National Laboratories, care au vizitat Japonia în 1982, a afirmat în raportul său că sistemele de calcul de mare capacitate din fabricația actuală a unor firme japoneze sînt foarte aproape de cele mai bune pe plan mondial.

Cu toate că în raport cu aceste proiecte japoneze, deosebit de ambițioase, ce reclamă eforturi materiale, financiare, umane și organizatorice ieșite din comun, s-au exprimat și rezerve și chiar unele critici de ordin tehnic. Cu toate că încă nu este evident modul în care vor fi soluționate și atinse multe din obiectivele acestor proiecte, toți specialiștii sînt însă de acord că, chiar dacă japonezii nu vor atinge decât parțial obiectivele pe care și le-au propus, ei vor marca un avans net în raport cu celelalte țări, inclusiv față de S.U.A., în domeniul tehnologiei calculatoarelor.

Referitor la problema factorilor care au influențat abordarea unui asemenea proiect gigant pentru prima oară în Japonia și nu într-o altă țară, se afirmă destul de frecvent că proiectul japonez pentru calculatorul de generația a cincea este un exemplu excelent de strategie economică. Japonezii au efectuat studii serioase înainte de lansarea proiectului, în care au analizat cerințele sociale și economice atât interne cît și internaționale la nivelul anilor '90.

După cum se știe, Japonia este deficitară din punct de vedere al teritoriului pe care îl ocupă, densitatea populației fiind foarte mare (circa de 40 de ori mai mare decât în S.U.A.); este, de asemenea, deficitară din punct de vedere al resurselor de materii prime, neputîndu-și asigura din resursele interne decât circa 15% din energia necesară, circa 0,3% din petrolul necesar și nici agricultura nu-i poate asigura integral necesitățile de alimente. În schimb dispune de o foarte valoroasă resursă umană, cu o educație și pregătire profesională de foarte bună calitate, foarte sîrguin-



cioasă, perseverentă și disciplinată, resursă pe care și-au propus să o valorifice la un nivel calitativ superior.

Pe de altă parte, Japonia este considerată pe ansamblu ca una din primele trei puteri economice mondiale, iar în unele domenii este pe primul loc în lume ca nivel atins de productivitate, de calitate și de cost în realizarea produselor industriale. În această situație, Japonia nu mai are sens să-și propună să ajungă din urmă alte țări mai dezvoltate, în schimb este logic să-și propună obiective de a deveni lider mondial și în domenii ca cercetarea științifică și dezvoltarea tehnologică de avangardă și în special în domenii de vîrf. Dintre aceste domenii de vîrf, japonezii au ales și industria de calculatoare datorită implicațiilor multiple și profunde pe care s-a dovedit că o are utilizarea calculatoarelor în creșterea semnificativă a productivității muncii în cele mai diferite domenii și mai ales a domeniilor în care cunoașterea, competența sînt esențiale („Knowledge intensive”).

Japonezii, conștienți de necesitatea de a rămîne competitivi pe piața mondială, și-au propus să realizeze acest obiectiv prin creșterea productivității în sectoare neglijate pînă în prezent din acest punct de vedere. Japonezii în sectoarele industriale primare cum sînt agricultura, pescuitul, și mai ales în sectoarele industriale terțiare cum sînt de exemplu, serviciile, cercetarea-proiectarea, managementul, medicina, educația ș.a. vor exploata cît mai mult cunoștințele și prin aceasta să devină cît mai productive. În ceea ce privește sectoarele industriale secundare (cele de prelucrare) japonezii și-au propus ca produsele lor să fie în continuare superioare calitativ, pe baza cunoștințelor expert care vor fi folosite în proiectarea și fabricarea acestor produse.

### 3. Scurt istoric

Ministerul japonez pentru Industrie și Cooperare Internațională (MITI), ca organ guvernamental preocupat de perspectivele pe termen lung ale industriei japoneze, a optat încă în anii '70 pentru orientarea economiei japoneze spre o societate informatizată. În realitate, această decizie luată de MITI a fost doar o consecință a unei decizii guvernamentale mai globale de a îndrepta societatea japoneză în această direcție, hotărîre ce implica în afară de MITI și alte departamente cum sînt Ministerul Sănătății și al Nivelului de Trai și Agenția de Planificare Economică și Ministerul Poștelor și Telecomunicațiilor. Există programe adecvate coordonate de fiecare din aceste organisme, proiectul generația a cincea de calculatoare urmărind să rezolve o serie de probleme importante în toate aceste programe.

MITI a însărcinat în anul 1978 Laboratorul Național Electrotehnic Japonez de a elabora un proiect de realizare a unui nou calculator pentru anii '90. De asemenea, și-a însușit primul raport întocmit de acest laborator asupra calculatorului de generația a cincea și a patronat o primă conferință internațională pe acest subiect, care a avut loc în octombrie 1981, și cu care ocazie s-a lansat public proiectul național japonez pentru generația a cincea, urmărindu-se să se obțină și cît mai multe păreri și reacții din partea unor specialiști competenți din Japonia și unele țări avansate (S.U.A., Anglia, R.F.G., Franța), invitați în mod special. După circa 6 luni, în aprilie 1982 se înființează Institutul pentru tehnologia



calculatoarelor de noua generație (ICOT) și la acest program aderă (cu suport uman, material și financiar) 8 firme japoneze importante (Fujitsu, Hitachi, Nippon Electric Corp., Mitsubishi, Matsushita, Oki, Sharp și Toshiba) precum și două laboratoare naționale: Laboratorul Musashino al Companiei japoneze de telefoane și telegraf (deținut de stat) și Laboratorul Electrotehnic (subordonat MITI).

Finanțarea proiectului japonez este evident substanțială, dar nu exagerat de mare în comparație cu bugetele unor firme sau proiecte americane. Astfel, bugetul pentru cei 10 ani cât este planificat proiectul prevede că se va consuma circa 1 miliard \$ (comparativ, de exemplu, cu bugetul de circa 1,2 miliarde \$ pentru activitatea de cercetare dezvoltare al firmei IBM pe anul 1982). Din bugetul total de circa 1 miliard \$ al proiectului japonez, guvernul asigură circa 450 milioane \$, în special în primii 3 ani ai proiectului, diferența fiind finanțare asigurată de firmele industriale interesate, participante la proiect, în funcție de succesele ce se vor realiza pe parcursul proiectului. Dintre firmele japoneze industriale deja implicate în anumite probleme aplicative concrete legate de generația a cincea de calculatoare, se citează: firma NEC interesată în realizarea de mașini tip PROLOG; Musashino Laboratory al Companiei japoneze de telefoane și telegraf în realizarea unei mașini LISP performante; Hitachi și Fujitsu în realizarea și utilizarea de sisteme expert (Fujitsu realizând deja o mașină LISP ce se atașează calculatoarelor universale Fujitsu).

#### **4. Domenii de aplicare pentru calculatoarele de generația a cincea**

Calculatoarele din generația a cincea propuse de japonezi, fiind definite ca sisteme de prelucrare a cunoștințelor, sînt sisteme complexe, dedicate a fi folosite, tot conform proiectului japonez [2], în domenii ca: sisteme inteligente de cercetare/proiectare asistată de calculator — CPAC (CAE/CAD = Computer Aided Engineering/Computer Aided Design); sisteme inteligente de instruire asistată de calculator — IAC (CAI — Computer Aided Instruction); sisteme inteligente de automatizare a activităților de birou (OA = Office Automation); roboți inteligenți — RI.

Deci este evident că, calculatoarele de generația a cincea nu vor înlocui treptat calculatoarele din generațiile precedente decît în unele tipuri de aplicații, deschizînd totodată perspectiva unor noi aplicații importante.

#### **5. Obiective pentru generația a cincea de calculatoare**

Din studiul preliminar japonez pentru generația a cincea de calculatoare se pot sintetiza următoarele 4 obiective de bază urmărite [2, 3, 5]:

(a) Creșterea substanțială a nivelului de inteligență cu care calculatoarele prelucrează informațiile precum și a capacității acestora de a comunica cît mai inteligent cu utilizatorul.

Ca și în cazul simțurilor umane care își pot realiza funcțiunile datorită existenței de cunoștințe necesare înțelegerii informațiilor percepute, pentru ridicarea nivelului de inteligență a calculatoarelor, inclusiv de



cooperare inteligentă cu utilizatorii umani, este absolut necesar ca aceste calculatoare să se sprijine în funcționarea lor pe cunoștințele specifice domeniului de utilizare. Pentru utilizarea eficientă a acestor cunoștințe, calculatorul trebuie să fie capabil să efectueze *inferențe logice* precum și funcții de *acumulare a cunoștințelor*.

(b) Capacitatea de a asista omul în acțiuni complexe cum ar fi explorarea de noi cunoștințe. În acest sens, inteligența calculatoarelor trebuie ridicată pînă la nivelul la care ele să poată înțelege mediul în care lucrează, în special modificările care se produc. Acest obiectiv presupune o funcționare a calculatorului în timp real, cu prelucrări paralele.

(c) Posibilitatea reprezentării informațiilor sub diverse forme, după necesități.

Față de varietatea, volumul și formele foarte limitate de reprezentare a informațiilor cu care operează calculatoarele actuale, japonezii și-au propus o lărgire considerabilă a acestor forme și volume, mai aproape de dimensiunile reale ale acestor probleme.

(d) Posibilitatea de achiziționare de noi cunoștințe prin simularea unor situații noi, necunoscute.

Se speră că se vor putea obține noi cunoștințe despre situații necunoscute a priori, pe baza unor simulări la scară mare într-o varietate de domenii cum ar fi, știința și tehnologia, managementul și societatea în general. Prin utilizarea de calculatoare super-rapide, se vor putea realiza simulări pînă în prezent imposibil de realizat.

Din punctul de vedere al utilizatorilor, japonezii propun următoarele funcțiuni pentru calculatorul de generația a cincea :

(a) Posibilitatea unei utilizări facile a acestor sisteme, fără a presupune cunoștințe profesionale. În acest scop, aceste sisteme vor poseda :

— funcții de introducere/extragere a informațiilor cu ajutorul imaginilor, a graficelor, a vorbirii și a limbajului natural ;

— funcții de prelucrare în regim conversațional a informațiilor reprezentate sub forme mai accesibile omului ;

— funcții de memorare a cunoștințelor atît generale cît și speciale tipului și domeniului de aplicație.

(b) Funcții de raționament și decizie similare cu cele umane. În acest scop, în proiectul japonez se prevăd :

— funcții care să permită regăsirea automată, dintr-un volum vast de informații memorate, a informațiilor pertinente, ca răspuns la întrebările formulate ;

— funcții care să permită, pentru probleme noi, necunoscute, generarea de concluzii pe bază de inferențe aplicate pe informații memorate ;

— funcții care să asigure învățarea și memorarea de cunoștințe ce vor fi utilizate la soluționarea viitoarelor noi probleme.

(c) Funcții care să faciliteze programarea. Japonezii propun o utilizare cît mai eficientă a software-ului ce se acumulează precum și o creștere a productivității de programare, prin :

— funcții care să permită generarea și modificarea de programe de însoși calculatorul ;

— funcții care să permită calculatorului să efectueze singur prelucrări și raționamente uzuale, de bun simț, fără a fi obligatoriu instruit de om.

(d) Funcții de configurare flexibilă și fiabilă, funcție de aplicație în sensul :

— configurării sistemului după aplicație ;



- posibilității extinderii modulare a sistemului;
- posibilității prelucrării distribuite;
- posibilității restabilirii automate după defecțiuni, cu pierderi minime, precum și facilități de auto-verificare și diagnosticare;
- posibilității secretizării fiabile a informațiilor.

## 6. Funcții de bază

Japonezii au propus trei funcții de bază pentru calculatorul de generația a cincea și anume:

- Funcții de inferență și rezolvare de probleme.
- Funcții de gestiune a cunoștințelor.
- Funcții de interfață inteligentă om-calculator.

Aceste funcții de bază se vor realiza atât prin hardware cât și prin software, urmărindu-se următoarele performanțe:

— Funcțiile de inferență și rezolvare de probleme se vor executa cu o viteză de ordinul a  $100 \text{ M}^1 - 1\text{G}^2$  LIPS ( $1 \text{ LIPS} =$  o inferență logică pe secundă; pentru a efectua 1 LIPS, calculatoarele actuale execută circa  $100 \div 1\,000$  secvențe sau pași de programe, deci 1 LIPS este echivalent cu circa  $100 \div 1\,000$  instrucțiuni pe secundă; deci calculatoarele actuale au o viteză de circa  $10^4 \div 10^5$  LIPS față de  $10^8 \div 10^9$  LIPS, deci cu circa patru ordine de mărime mai mare propus pentru generația a cincea).

— Funcția de gestiune a cunoștințelor va avea performanța de a regăsi în câteva secunde cunoștințele necesare unei inferențe, dintr-o bază de cunoștințe cu capacitatea de maximum  $100 - 1\,000$  Gbyte.

— Interfața inteligentă cu un asemenea sistem (a 3-a funcție de bază a calculatorului de generația a cincea) va asigura comunicarea cu ajutorul vorbirii, a imaginilor, a limbajului natural.

Funcțiile de bază precum și performanțele calculatoarelor de generația a cincea, conform proiectului japonez, vor fi suficient de generale și corespunzătoare pentru utilizarea lor în traducerea automată, sisteme întrebare-răspuns folosind vorbirea și imaginile, aplicații ce vor fi destul de frecvente în anii '90.

## 7. Imaginea conceptuală din punct de vedere al utilizatorului

Un exemplu de calculator de generația a cincea, după proiectul japonez, este ilustrat în figura 1.

După cum s-a mai arătat, calculatoarele din generația a cincea concepute de japonezi vor fi orientate către prelucrarea cunoștințelor, având capabilități puternice de prelucrare logică a informațiilor.

Una din trăsăturile definitorii ale unui asemenea calculator este interfața dintre utilizator și calculator, care se va apropia foarte mult de nivelul posibilităților umane. Dacă până în prezent, la generațiile anterioare de calculatoare, comunicarea om-calculator este posibilă doar prin intermediul programelor, rezolvarea unei probleme cu ajutorul calculatorului necesitând elaborarea de către utilizator a unui program adecvat

<sup>1)</sup>  $1\text{M} = 10^6$

<sup>2)</sup>  $1\text{G} = 10^9$ .



problemei, folosind anumite limbaje de programare procedurale, în cazul generației a cincea de calculatoare, descrierea problemei de rezolvat, inclusiv modelarea ei, va avea loc la nivelul interfeței de dialog cu utilizatorul. Cu alte cuvinte, calculatorul va fi capabil să înțeleagă descrierea unei probleme de rezolvat, să construiască pentru aceasta un model și să genereze un program adecvat. Comunicarea om-calculator va fi posibilă prin vorbire, limbaj natural, grafică și chiar imagini.

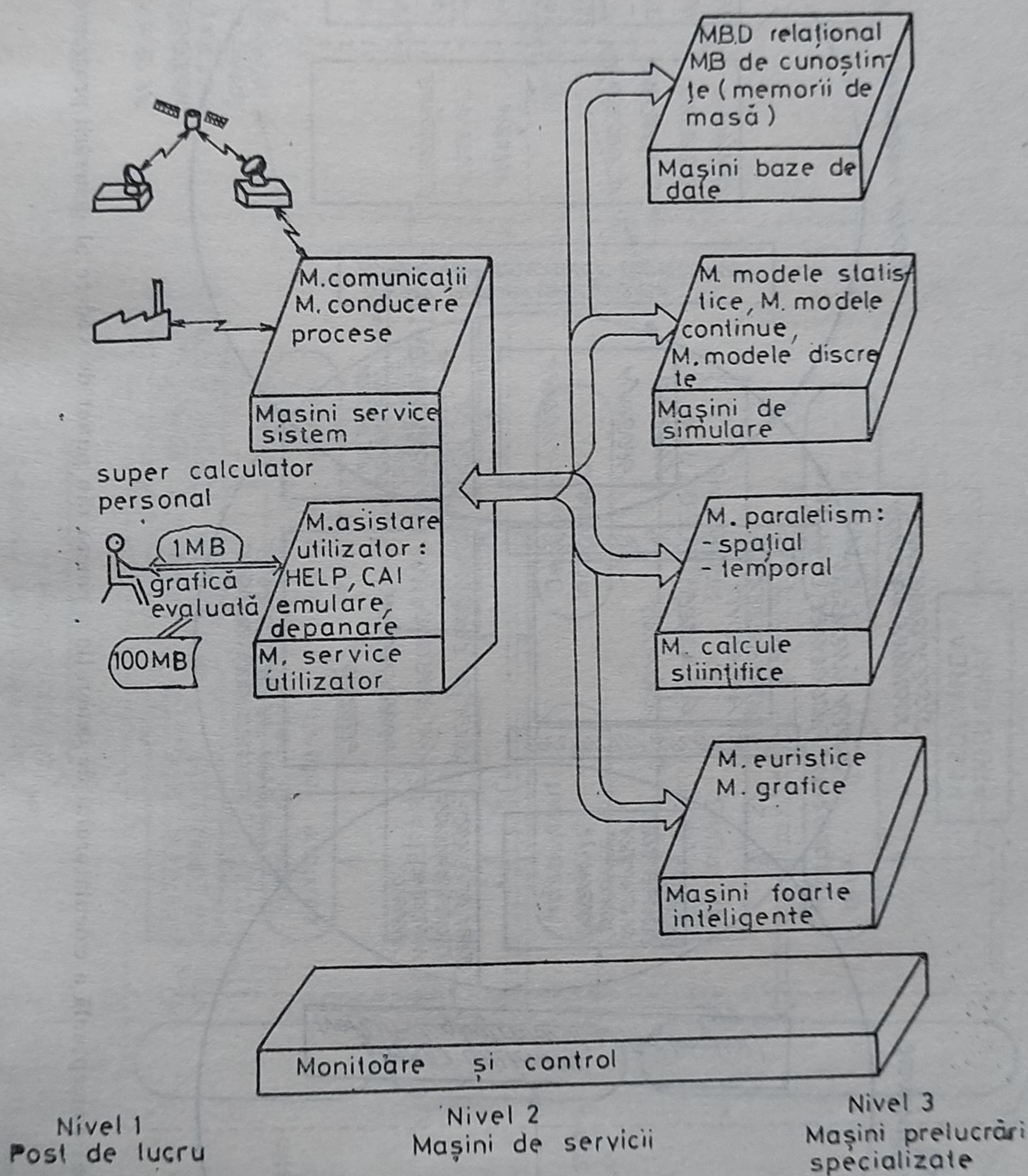


Fig. 1. — Un model de calculator de generația a cincea.

În figura 2 se reprezintă conceptual imaginea unui calculator de generația a cincea din punctul de vedere al programării aplicațiilor. Din figură se poate constata că funcțiunile cablate sînt de un nivel mult mai ridicat decît în cazul calculatoarelor actuale. Sistemul comunicare om-calculator și de elaborare a programelor aplicative este el însuși un sistem sofisticat de prelucrare a cunoștințelor, ce realizează înțelegerea problemei



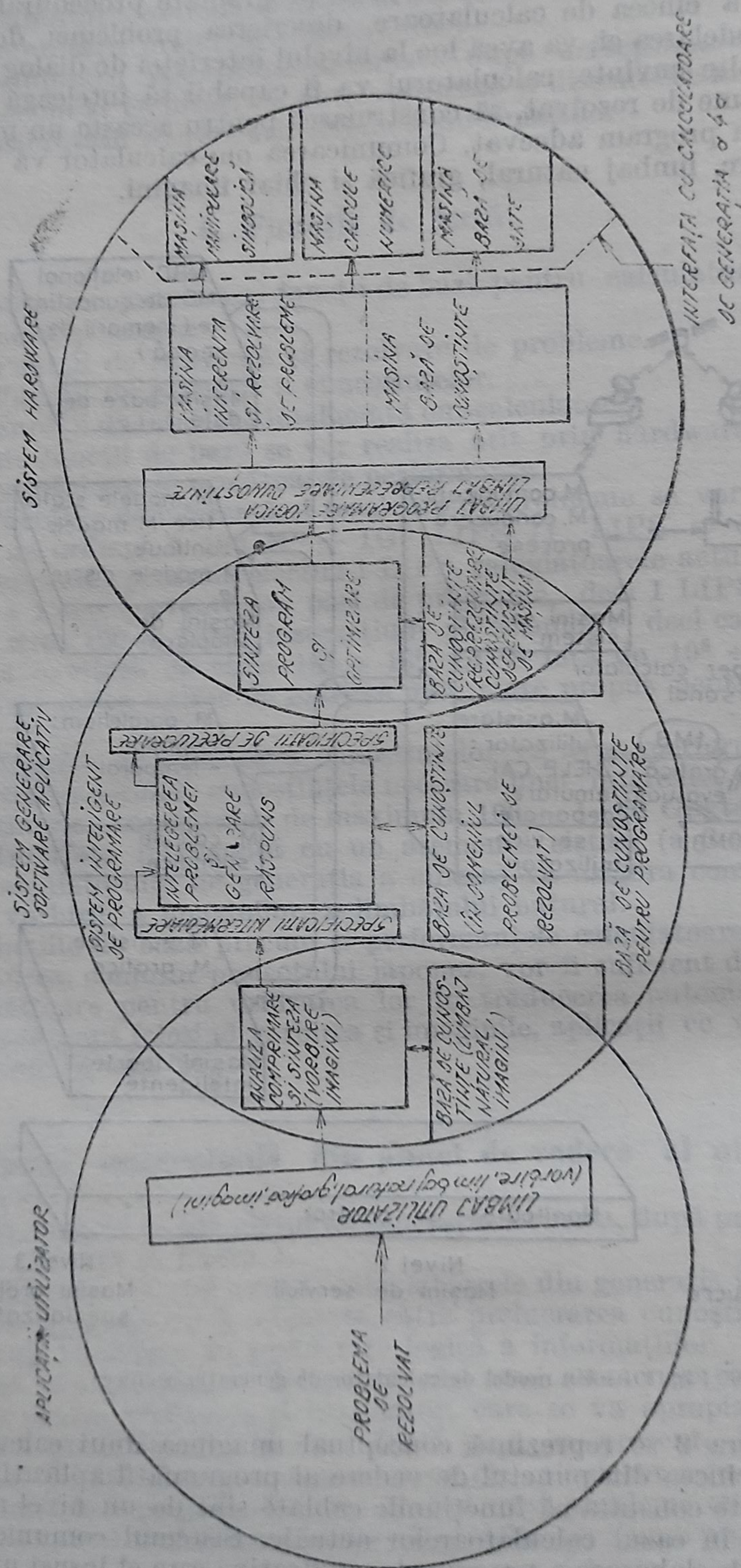


Fig. 2. - Imaginea conceptuală a calculatorului de generația a cincea din punct de vedere al generării programelor aplicative.



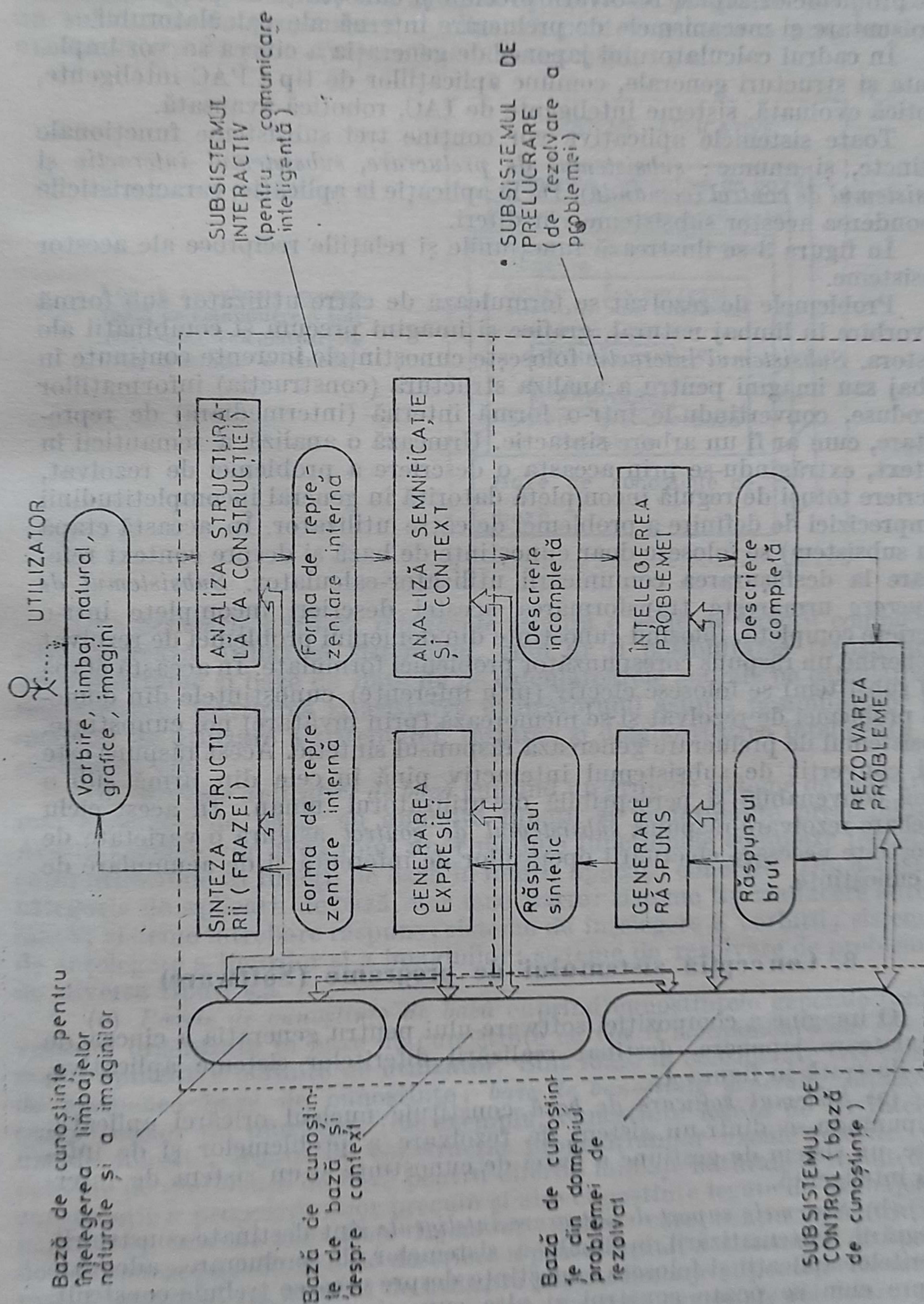


Fig. 3. — Funcțiile și relațiile reciproce ale celor trei subsisteme funcționale ale unui calculator de generația a cincea.



și sintetizarea programelor. Baza de cunoștințe a acestui subsistem se referă la cunoștințe despre limbajele naturale, despre imagini, despre domeniile problemelor supuse rezolvării precum și cunoștințe despre formele de reprezentare și mecanismele de prelucrare internă ale calculatorului.

În cadrul calculatorului japonez de generația a cincea se vor implementa și structuri generale, comune aplicațiilor de tip CPAC inteligente, birotică evoluată, sisteme inteligente de IAC, robotică avansată.

Toate sistemele aplicative vor conține trei subsisteme funcționale distincte, și anume: *subsistemul de prelucrare*, *subsistemul interactiv* și *subsistemul de control (comandă)*. De la aplicație la aplicație, caracteristicile și ponderea acestor subsisteme vor diferi.

În figura 3 se ilustrează funcțiunile și relațiile reciproce ale acestor subsisteme.

Problemele de rezolvat se formulează de către utilizator sub formă de vorbire în limbaj natural, grafice și imagini precum și combinații ale acestora. *Subsistemul interactiv* folosește cunoștințele inerente conținute în limbaj sau imagini pentru a analiza structura (construcția) informațiilor introduse, convertindu-le într-o formă internă (intermediară) de reprezentare, cum ar fi un arbore sintactic. Urmează o analiză a semanticii în context, extrăgându-se prin aceasta o descriere a problemei de rezolvat, descriere totuși de regulă incompletă datorită în general incompletitudinii și impreciziei de definire a problemei de către utilizator. În această etapă (sau subsistem) se folosesc doar cunoștințe de bază și despre context referitoare la desfășurarea comunicării utilizator-calculator. *Subsistemul de prelucrare* urmărește transformarea acestei descrieri incomplete într-o descriere completă, folosind cunoștințe din domeniul problemei de rezolvat și generând un răspuns corespunzător problemei formulate. În această etapă (sau subsistem) se folosesc efectiv (prin inferențe) cunoștințele din domeniul problemei de rezolvat și se memorează (prin învățare) noi cunoștințe. Subsistemul de prelucrare generează răspunsul sintetic. Acest răspuns este apoi convertit de subsistemul interactiv pînă în cele din urmă sub o formă convenabilă și perceptibilă de utilizatorul uman. În acest ciclu întrebare-rezolvare-răspuns, *subsistemul de control* asigură o varietate de cunoștințe necesare efectuării operațiilor de inferență și de acumulare de noi cunoștințe.

## 8. Concepția sistemului de programe (Software)

O imagine a compoziției software-ului pentru generația a cincea de calculatoare japoneze, destinat realizării diferitelor sisteme aplicative, este ilustrată în figura 4.

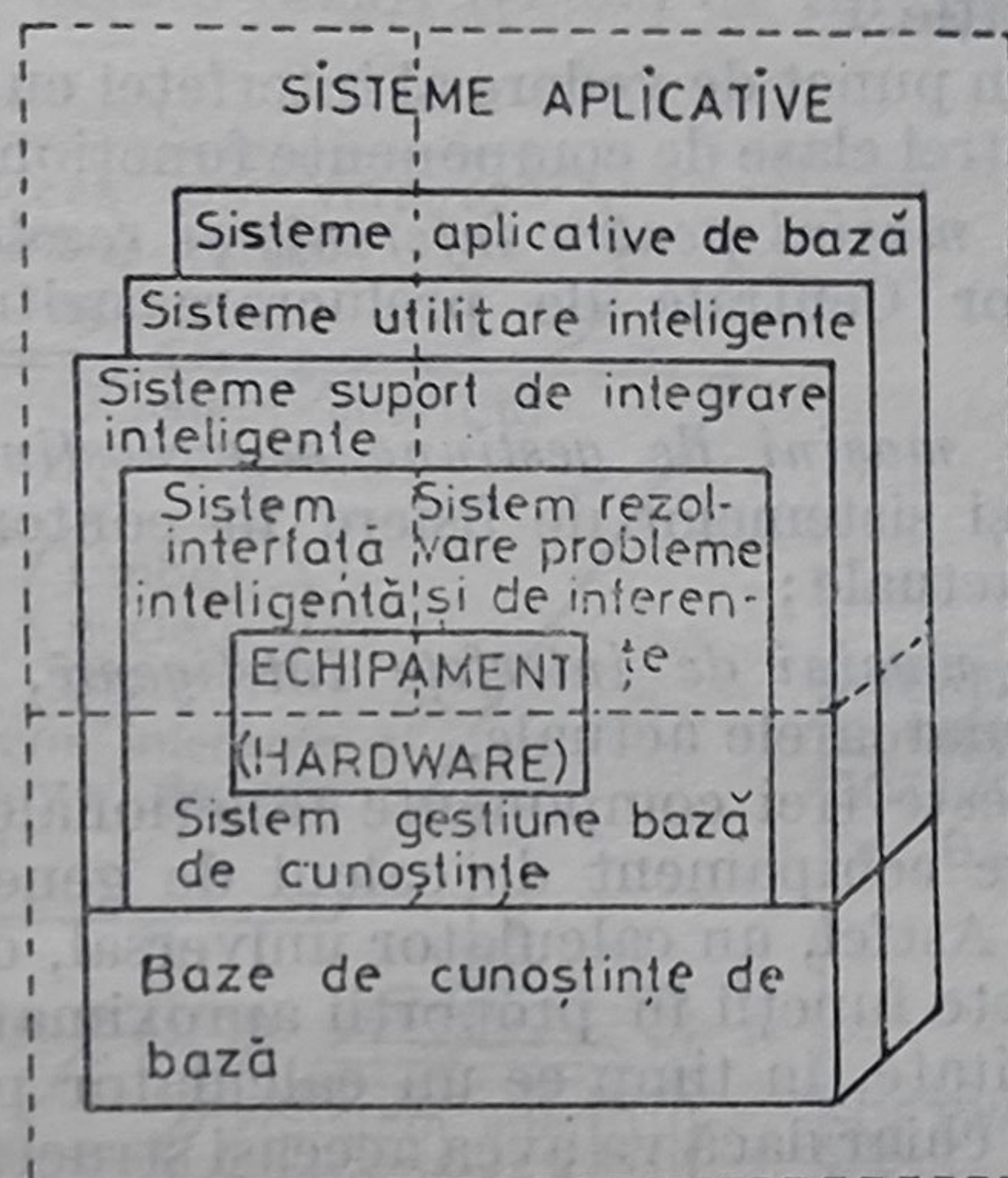
(a) *Sistemul software de bază* constituie nucleul oricărei aplicații, compunându-se dintr-un sistem de rezolvare a problemelor și de inferențe, un sistem de gestiune a bazei de cunoștințe și un sistem de interfață inteligentă.

(b) *Sistemele suport de integrare inteligente* sînt destinate construirii, integrării, sistematizării optimale a sistemelor de prelucrare, adecvate diferitelor aplicații și folosesc cunoștințe despre ceea ce trebuie construit, despre cum se poate construi și alte cunoștințe similare, reducînd la minim ceea ce omul trebuie să sistematizeze. Aceste sisteme cuprind subsisteme ce construiesc procesul de prelucrare adecvat, pornind de la



specificațiile de descriere (ce folosesc un limbaj de descriere a specificațiilor), un subsistem de verificare a corectitudinii și un subsistem pentru simularea operațiilor etc. Din această categorie de programe fac parte și un subsistem de programare inteligentă pentru generarea programelor, un subsistem de proiectare de baze de cunoștințe ș.a.

Fig. 4. — Schema conceptuală de compunere a software-ului calculatoarelor de generația a cincea.



(c) *Sistemele utilitare inteligente* asigură o serie de funcții complexe care ușurează utilizarea sistemului, cum ar fi posibilitatea transferării unor programe și baze de date de pe calculatoarele actuale pe calculatoarele viitoare, instruirea și asistarea utilizatorului asupra posibilităților și modului de utilizare a sistemului, testarea și diagnosticarea inteligentă a sistemului etc.

(d) *Sisteme aplicative de bază* cuprind un grup de sisteme de aplicații mai generale, de bază, cu anumite funcții și performanțe prestabilite. Aceste sisteme vor avea rolul unor module funcționale complexe (sophisticate) utilizabile ca funcții de bază în multe aplicații concrete. Din această categorie de aplicații de bază, vor face parte: sisteme tip traducere automată; sisteme întrebare-răspuns; sisteme de înțelegere a vorbirii; sisteme de înțelegere a figurilor și a imaginilor; sisteme de rezolvare de probleme de diverse tipuri ș.a.

(e) *Bazele de cunoștințe de bază* cuprind cunoștințele generale (universale) organizate în baze de cunoștințe ce vor fi componente ale sistemelor aplicative definite de utilizator. Sînt luate în considerare trei clase de asemenea baze de cunoștințe; *baze de cunoștințe generale*, similare cunoștințelor de bun simț (de exemplu, cunoștințe legate de cuvintele uzuale folosite, legate de construcția propozițiilor uzuale folosite, o colecție de dicționare de bază pentru diferite limbaje naturale și reguli de construcție a propozițiilor precum și alte cunoștințe legate de limbajele naturale); *baze de cunoștințe legate de sistem* care conțin specificații despre sistem (specificații de descriere a procesorului, a sistemului de operare, specificații tip manual de utilizare pentru limbaje și pentru cele mai uzuale programe cu caracter utilitar etc.); *baze de cunoștințe din domeniul aplicației* ce conțin cunoștințe din anumite domenii foarte generale de utilizare a sistemului (de exemplu, în domeniul CPAC, CAI etc.).



## 9. Concepția structurii hardware

Conform proiectului japonez, generația a cincea de calculatoare va acoperi o gamă foarte largă de echipamente, de la calculatoare personale profesionale la sisteme de calcul de mari proporții, ce vor fi utilizate funcție de aplicație.

Din punct de vedere al interfeței cu sistemul de operare, hardware-ul va avea trei clase de componente funcționale și anume :

(a) *mașini pentru inferențe și rezolvarea de probleme*, corespunzător Unităților Centrale de prelucrare aritmetico-logică din calculatoarele actuale ;

(b) *mașini de gestiune a cunoștințelor*, corespunzător memoriilor interne și sistemelor de fișiere în context memorie virtuală din calculatoarele actuale ;

(c) *mașini de interfețe inteligente*, corespunzător canalelor de I/E din calculatoarele actuale.

Aceste trei componente funcționale hardware se vor regăsi practic în fiecare echipament de calcul de generația a cincea, dar în proporții diferite. Astfel, un calculator universal, de putere medie, va avea dezvoltate aceste funcții în proporții aproximativ egale și la un nivel mediu de complexitate, în timp ce un calculator personal profesional de generația a cincea, chiar dacă va avea aceeași structură hardware, funcțiile respective vor fi mai puțin dezvoltate ca funcții hardware.

Sistemele de calcul de generația a cincea cu funcția hardware pentru inferențe și rezolvarea de probleme foarte dezvoltată, vor fi destinate aplicațiilor ce necesită cunoștințe profesionale tip expert și volum mare de inferențe. Sistemele ce vor avea dezvoltate preponderent funcția hardware de gestiune a bazelor de cunoștințe vor fi adevărate „calculatoare de gestiune a cunoștințelor”, fiind utilizate în domenii care presupun memorarea unor cunoștințe de volum foarte mare. Sistemele de calcul ce vor avea dezvoltate preponderent funcții de interfațare inteligente vor dispune de dispozitive periferice pentru intrare/ieșire tip vorbire și tip imagini.

Va fi desigur posibilă utilizarea acestor tipuri de calculatoare de generația a cincea atât independent cât și în combinații.

În figura 5 se ilustrează configurația de bază hardware și software a unui calculator din generația a cincea.

Arhitectura hardware va fi bazată pe șase mașini funcționale, ce vor fi combinate într-o structură distribuită funcțional, folosind tehnici de modularizare și microprogramare.

În cazul calculatoarelor de putere și performanță moderată, se va adopta o arhitectură bazată pe multă microprogramare (firmware). Vor fi implementate limbaje bazate pe *logica predicatelor* și *date de tip abstract*.

Pentru sistemele de calcul de mare capacitate și performanțe ridicate, se vor folosi mașini tip „*data flow*” = „*flux de date*”; astfel, pentru sistemele de inferențe și rezolvări de probleme, partea de execuție a mașinilor cu programare logică va folosi mecanisme performante de tip „*data flow*”, iar baza de cunoștințe va fi gestionată de o mașină ultra-rapidă de algebră relațională (care la rândul ei, pentru execuție, va folosi un mecanism adecvat tip „*data flow*”).

Un sistem de calcul puternic pentru gestiunea de baze de cunoștințe va folosi ca nucleu o mașină bază de date relațională performantă,



incluzînd şi o maşină bazată pe algebra relaţională şi eventual şi maşini utilitare bazate pe date de tip abstract.

Partea de hardware pentru sistemul de interfeţe inteligente va cuprinde şi procesoare VLSI specializate pentru prelucrarea vorbirii şi respectiv a imaginilor. Pentru operaţiunile care trebuie executate cu foarte mare viteză, se vor folosi şi în aceste cazuri tehnici de tip „data flow”.

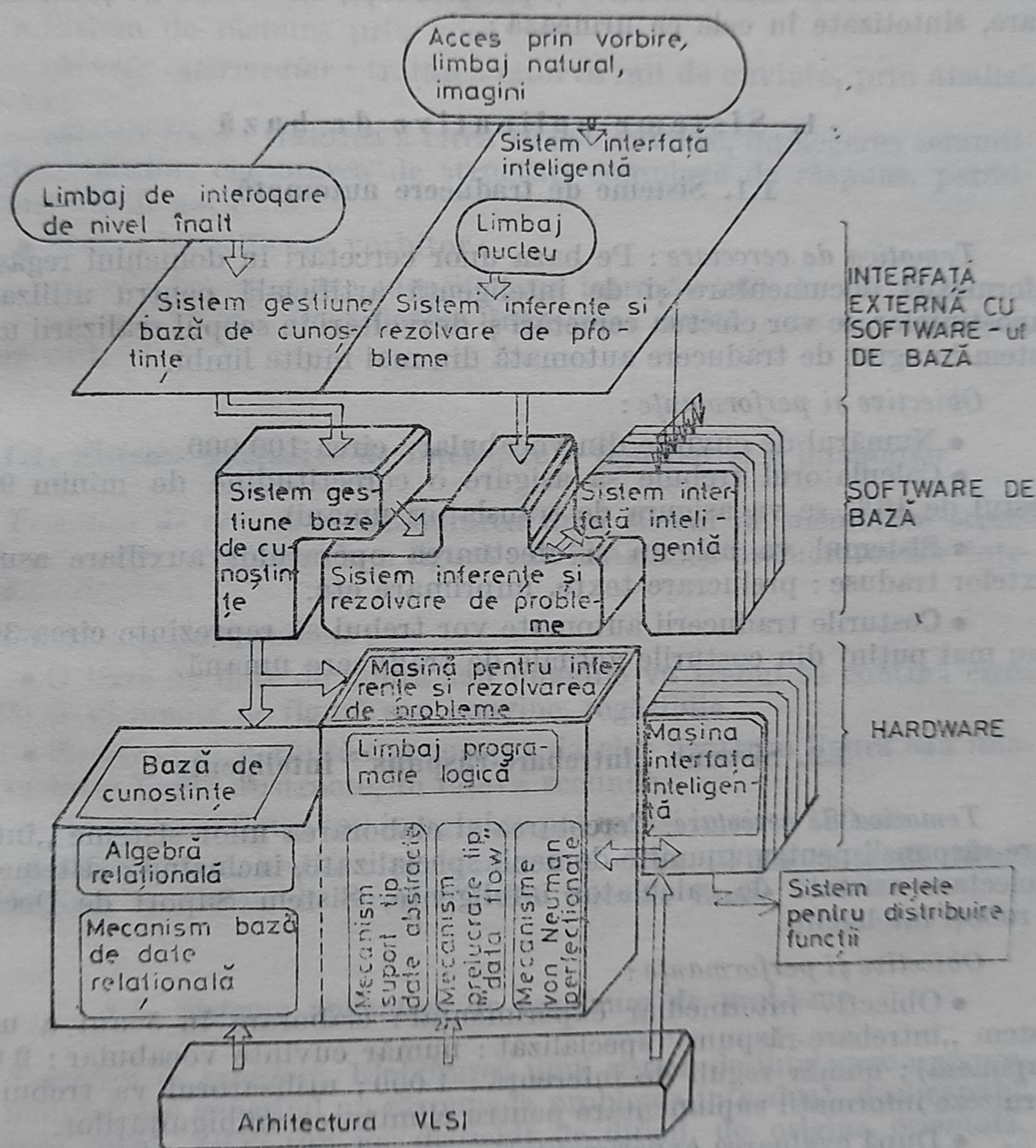


Fig. 5. — Configuraţia de bază a unui sistem de calcul de generaţia a cincea.

După cum se vede, maşinile de tip „data flow” vor constitui mecanismul hardware de bază pentru prelucrarea cu foarte mare viteză, de unde şi importanţa acordată studierii acestor maşini în cadrul programului japonez. Avînd în vedere că pentru atingerea unor performanţe acceptabile proiectul japonez acordă o mare importanţă şi sistemelor de proiectare asistată de calculator a circuitelor VLSI.



## 10. Principalele obiective de cercetare-dezvoltare din proiectul japonez

Temele de cercetare-dezvoltare în cadrul proiectului japonez sînt grupate în şapte domenii de bază şi în cadrul lor pe mai multe direcţii mai importante; pentru fiecare din aceste domenii şi teme de cercetare-dezvoltare sînt stabilite obiective şi performanţe, atît finale cît şi intermediare, sintetizate în cele ce urmează :

### 1. Sisteme aplicative de bază

#### 1.1. Sisteme de traducere automată

*Tematica de cercetare :* Pe baza unor cercetări în domeniul regăsirii informaţiei documentare şi de inteligenţă artificială pentru utilizarea cunoştinţelor, se vor efectua cercetări şi dezvoltări în scopul realizării unui sistem integrat de traducere automată din mai multe limbi.

*Obiective şi performanţe :*

- Numărul de cuvinte din vocabular : circa 100 000.
- Calculatorul trebuie să asigure o corectitudine de minim 90 % (restul de 10 % se va asigura de traducători umani).
- Sistemul va asigura şi efectuarea operaţiilor auxiliare asupra textelor traduse : prelucrare texte, imprimare etc.
- Costurile traducerii automate vor trebui să reprezinte circa 30 % (sau mai puţin) din costurile actuale de traducere umană.

#### 1.2. Sisteme „întrebare-răspuns” inteligente

*Tematica de cercetare.* Cercetarea şi elaborarea unor sisteme „întrebare-răspuns” pentru anumite domenii specializate, incluzînd : sisteme de proiectare asistată de calculator inteligente, Sistem Suport de Decizie şi roboţi inteligenţi.

*Obiective şi performanţe :*

- Obiectiv intermediar experimental : elaborare în 5 ani a unui sistem „întrebare-răspuns” specializat : număr cuvinte vocabular : 2 000 (japoneză) ; număr reguli de inferenţă : 1 000 ; utilizatorul va trebui să furnizeze informaţii suplimentare pentru eliminarea ambiguităţilor.
- După evaluarea experimentală, se va elabora o aplicaţie prototip pe domenii specializate : număr cuvinte vocabular : peste 5 000 ; număr reguli de inferenţă : peste 10 000.

#### 1.3. Sisteme aplicative de înţelegere a vorbirii

*Tematica de cercetare.* Cercetare şi elaborare a unui sistem de identificare a vorbitorului ca parte componentă a unui sistem universal de răspuns prin voce, ca sistem de intrare-ieşire pentru o maşină de scris fonetică şi pentru un sistem de informare telefonică.



#### *Obiective și performanțe :*

- Mașina de scris fonetică :
  - *obiectiv intermediar* : un sistem cu construcții simple a frazelor, capabil să manipuleze de la câteva sute la câteva mii de cuvinte ;
  - *obiectiv final* : manipularea a circa 10 000 cuvinte cu analiza semantică simultană, corecția automată a erorilor în timpul recunoașterii vorbirii și generarea de fraze inteligente.
- Sistem de răspuns prin voce :
  - *obiectiv intermediar* : tratarea câtorva mii de cuvinte, prin analiză și sinteză ;
  - *obiectiv final* : tratarea a circa 10 000 cuvinte, înțelegerea semnificației întrebărilor, elaborarea de structuri complexe de răspuns, permițând conversații naturale.
- Sistem identificare vorbitor :
  - *obiectiv intermediar* : identificarea între 50÷60 de vorbitori ;
  - *obiectiv final* : identificarea între mai multe sute de vorbitori, în timp util.

### **1.4. Sisteme aplicative de înțelegere a figurilor și a imaginilor**

*Tematica de cercetare.* Elaborarea unui sistem de memorare structurală a datelor referitoare la o figură sau imagine și prelucrarea lor inteligentă.

#### *Obiective și performanțe :*

- O bază de date de figură sau imagine va trebui să conțină circa 100 000 de elemente de figură sau imagine, regăsibile.
- Sistemul să permită memorarea datelor pentru o figură sau imagine, inclusiv liniile abstracte, în câteva secunde.
- Datele unei figuri sau imagini se pot regăsi în circa 100 ms.
- *Obiectiv intermediar* : prelucrarea a circa 10 000 date de figură sau imagine, la viteza de circa 1/2 din vitezele finale.

### **1.5. Sisteme aplicative de rezolvare de probleme**

*Tematica de cercetare.* Elaborarea unui sistem de înțelegere a expresiilor matematice generând un răspuns la problema introdusă, deci capabil de a juca jocul „GO” (un joc deosebit de dificil, de origine orientală, ce constă în plasarea unor piese albe și negre într-o rețea de  $19 \times 19$  poziții [12]).

#### *Obiective și performanțe :*

- Sistem de înțelegere a expresiilor matematice :
  - *obiectiv intermediar* : sistem cu o bază de cunoștințe ce combină performanțele sistemului expert existent MAOSYMA cu funcții de prelucrare a inegalităților și ecuațiilor simple ;
  - *obiectiv final* : sistem de reprezentare a cunoștințelor și de rezolvare de probleme referitoare la expresii matematice ce combină algoritmi complecși de calcul.



- Sistem de joc „GO” :
  - *obiectiv intermediar* : sistem cu performanțe de joc echivalent unui jucător amator de nivel 10 ;
  - *obiectiv final* : sistem cu performanțe de joc echivalente unui jucător de nivel 1.

## 2. Sistemul software de bază

### 2.1. Sisteme de gestiune a bazelor de cunoștințe

*Tematica de cercetare.* Cercetarea și elaborarea unor tehnici de gestiune inteligente pentru reprezentarea și memorarea cunoștințelor umane în cadrul sistemelor, de calcul pentru utilizarea lor în sprijinul rezolvării de probleme.

*Obiective și performanțe :*

- Sistem de gestiune a bazelor de cunoștințe.
  - *Obiectiv intermediar* : gestiunea simultană a regulilor și a datelor, mecanism de acces optimal la baze de date, mecanism pentru eliminarea inconsistențelor, interfețe cu o mașină inferențială.
  - *Obiectiv final* : bază de cunoștințe din domenii diferite, bază de cunoștințe distribuită, învățare pe bază de inferențe inductive, integrarea cu o mașină inferențială.
- Mașină bază de cunoștințe :
  - *Obiectiv intermediar* : memorare și regăsirea a 2000 reguli și 1 000 000 elemente ( $10^3$  bait per element).
  - *Obiectiv final* : memorarea și regăsirea a 20 000 reguli și 100 000 000 elemente ( $\div 100$  GBait).

### 2.2. Sisteme de inferențe și rezolvare de probleme

*Tematica de cercetare.* Va constitui nucleul funcțional de prelucrare. Se va cerceta și elabora un sistem de rezolvare de probleme prin elaborarea unui model de prelucrare pentru rezolvarea de probleme și a unui sistem de inferențe.

*Obiective și performanțe :*

- Mașina inferențială :
  - *Obiectiv final* : circa  $10^2 \div 10^3$  Mega LIPS (vezi nota de subsol de la pag. 24).
- Limbajul de codificare pentru rezolvarea de probleme va trebui să suporte *programare funcțională și logică* precum și *programare modulară*, orientată pe obiecte. Modulele de program generate vor constitui o componentă software inteligentă (bazată pe cunoștințe) ce va servi sistemului de programare inteligentă.

### 2.3. Sisteme de interfețe inteligente

*Tematica de cercetare.* Cercetarea și elaborarea de tehnici pentru funcții de comunicare conversațională flexibilă pentru eliminarea barierelor de limbaj între calculator și utilizator.



#### *Obiective și performanțe :*

- Sistem de înțelegere a vorbirii și a limbajului natural :
  - identificarea semnalelor de tip vorbire, în timp real ;
  - sistem ce se poate adapta pentru a putea comunica cu orice vorbitor ;
  - vocabularul va cuprinde terminologia actuală, generală și specifică din cel puțin un domeniu concret de utilizare ;
  - sistemul va sintetiza cuvinte în limba japoneză și engleză.
- Sistem de înțelegere a figurilor și imaginilor :
  - figurile și imaginile tratate vor fi de complexitatea unor desene de organe de mașini și fotografii de uz medical (ca obiectiv intermediar, tratarea figurilor și imaginilor de complexitate circa 70 % din obiectivul final și viteză mai redusă).

### **3. Arhitecturi noi evolute**

#### **3.1. Mașini bazate pe programare logică**

*Tematica de cercetare.* Studii și elaborări de arhitecturi pentru realizarea de inferențe și a unui model de calcul bazat pe logica predicatelor cu o putere de exprimare apropiată de a limbajelor naturale.

#### **3.2. Mașini funcționale**

*Tematica de cercetare.* Elaborarea de arhitecturi pentru implementarea de modele funcționale și limbaje de programare adaptate prelucrării simbolice.

#### *Obiective și performanțe :*

- Mașină LISP personală, de  $2 \div 3$  ori mai performante decât un calculator universal (4 MIPS) în prelucrarea de liste.
- Mașini cu paralelism pronunțat, de 10 ori mai performante în prelucrarea de liste decât un calculator universal.
- Mașini „data flow” (tip flux de date), de câteva sute sau mii de ori mai performante pentru prelucrarea de liste decât un calculator universal.

#### **3.3. Mașini bazate pe algebra relațională**

*Tematica de cercetare.* Cercetări pentru elaborarea unei arhitecturi pentru efectuarea pe operații cu mulțimi, folosind ca limbaj de interfață algebra relațională (care va constitui nucleul sistemelor viitoare de baze de date).

#### *Obiective și performanțe :*

- Număr elemente de prelucrare în procesoarele paralele :

*Obiectiv intermediar :* cel mult o sută.

*Obiectiv final :* 500 + 600.



- Capacitate de memorare :

- memorie de capacitate mică pentru operații foarte rapide :  
 $10 \div 100$  Mbait;

- memorie de capacitate medie pentru operații rapide și medii :  
 $100 \text{ M} \div 10$  Gbait;

- memorie de capacitate mare pentru operații de viteză mică și medie :  $10 \div 1\,000$  Gbait.

### 3.4. Mașini utilitare pentru date de tip abstract

*Tematica de cercetare.* Cercetarea și elaborarea de structuri de memorare și funcții de prelucrare pentru modularizarea complexului software.

*Obiective și performanțe :*

- mașini bazate pe date de tip abstract cu circa 100 procesoare non-von Neumann paralele ;

- mașini bazate pe date de tip abstract cu circa 1 000 procesoare non-von Neumann paralele.

### 3.5. Mașini „data flow” (tip „flux de date”)

*Tematica de cercetare.* Cercetări privind arhitecturi pe modele tip „flux de date” („data flow”) orientate pe prelucrare paralelă și realizarea pe această bază a unor paralelisme sofisticate.

*Obiective și performanțe :*

- *Obiectiv inițial :* 16 procesoare cu o memorie de 8 Mbait (nivel operativ de bază).

- *Obiectiv intermediar :* 100 procesoare cu o memorie de 100 Mbait și viteză de circa 50 MIPS (de utilitate practică).

- Rețele de prelucrare cu  $10^3 \div 10^4$  procesoare.

- *Obiectiv final :* o mașină ultra rapidă tip „flux de date”, cu  $10^3 \div 10^4$  procesoare, memorie  $1 \div 10$  Gbait și viteză  $10^3 \div 10^4$  MIPS.

- Calculator personal tip „flux de date”, cu 32 procesoare, memorie de 10 Mbait și viteză 10 MIPS.

### 3.6. Mașini von Neumann perfecționate

*Tematica de cercetare.* Elaborarea de arhitecturi de mașini von Neumann perfecționate, reținând în special ceea ce este avantajos la aceste mașini și combinate cu tehnologii VLSI complexe.

*Obiective și performanțe :*

- *Obiectiv intermediar :* procesor tip von Neumann perfecționat cu densitatea de circa 1 milion tranzistori pe un cristal.

- *Obiectiv final :* procesor tip von Neumann perfecționat cu densitatea de circa 10 milioane tranzistori pe un cristal.



## 4. Arhitecturi distribuite funcțional

### 4.1. Arhitecturi de distribuire a funcțiilor

*Tematica de cercetare.* Elaborarea unei arhitecturi cu funcțiuni distribuite care să asigure intrinsec o înaltă eficiență, fiabilitate ridicată, construcție și utilizare simplă, adaptabilitate ușoară la viitoare perfecționări tehnologice.

### 4.2. Arhitecturi de rețele de calculatoare

*Tematica de cercetare.* Elaborarea de tehnice de cuplare a sistemelor de calcul în rețele globale și realizarea de sisteme cu prelucrare distribuite bazate pe rețele locale de mare viteză.

### 4.3. Mașini baze de date

*Tematica de cercetare.* Elaborarea unei mașini specializate cu o arhitectură adecvată pentru gestionarea bazelor de date și asigurarea unui acces rapid la baze de date de volum foarte mare.

*Obiective și performanțe:*

- Mașini experimentale: capacitate: până la 100 Gbait; viteză:  $10^3$  tranzacții/s; model date: relațional.
- Mașini utilizabile în practică: capacitate: până la 1000 Gbait; viteză:  $10^4$  tranzacții/s; model date: relațional.

### 4.4. Mașini pentru prelucrări numerice ultra rapide

*Tematica de cercetare.* Elaborarea unei mașini specializate pentru calcule numerice ultra rapide destinate simulărilor (care vor înlocui experimentele).

*Obiective și performanțe:*

- elaborarea unor elemente funcționale de calcul, utilizând noi componente de mare viteză ( $40 \div 100$  M op. virg. mobilă/s);
- elaborarea unei mașini de calcul paralele care va asigura funcționarea a circa 1 000 asemenea elemente funcționale în paralel, obținând o viteză globală de circa  $10^9$  op. virg. mobilă/s.

### 4.5. Sisteme de comunicare om-calculator de nivel înalt

*Obiective și performanțe:*

- Sistem de intrare/ieșire de caractere
- *Obiectiv intermediar:* un terminal display cu funcții de introducere a  $3\,000 \div 4\,000$  caractere în 4  $\div$  5 stiluri diferite de scriere.
- *Obiectiv final:* funcții suplimentare care să permită introducerea de caractere chinezești împreună cu vorbire, comprimare semantică.



- Sisteme intrare/ieșire de imagini :
  - *Obiectiv intermediar*: un dispozitiv de introducere tip tabletă în coordonate de  $5\,000 \times 5\,000 \div 10\,000 \times 10\,000$  puncte.
  - *Obiectiv final*: funcții inteligente mai avansate bazate pe specificațiile stabilite pentru tematica de cercetare-dezvoltare intitulată „Sisteme aplicative de înțelegere a figurilor și imaginilor”.
- Sistem de intrare/ieșire prin vorbire :
  - *Obiectiv intermediar*: capacitate de a identifica  $500 \div 1\,000$  cuvinte.
  - *Obiectiv final*: funcții inteligente mai avansate, bazate pe specificațiile stabilite pentru tematica de cercetare dezvoltare intitulată „Sisteme aplicative de înțelegere a vorbirii”, incluzând posibilități de comprimare semantică și de a trata limbaje parțial naturale.
- Terminal integrat cu funcții multiple de intrări/ieșiri. Toate funcțiunile de mai sus vor fi realizate și combinate pe bază de circuite VLSI pentru elaborarea de terminale tip calculator personal integrat.

## 5. Tehnologii VLSI

### 5.1. Arhitecturi VLSI

*Tematica de cercetare.* Elaborarea de arhitecturi care să exploateze caracteristicile unor VLSI cu densitatea de circa 1 milion tranzistoare pe un cristal (prevăzute a fi disponibile în jurul anul 1990).

*Obiective și performanțe:*

- *Obiectiv intermediar*: arhitectură completă pe un cristal cu o densitate de 1 milion tranzistori/cristal.
- *Obiectiv final*: arhitectură completă pe un cristal cu o densitate de 10 milioane tranzistori/cristal.

### 5.2. Sisteme inteligente de proiectare asistată de calculator a circuitelor VLSI

*Tematica de cercetare.* Elaborarea unui sistem integrat de proiectare asistată de calculator a circuitelor VLSI, capabile de a acumula cunoștințe pentru o utilizare cât mai eficientă.

*Obiective și performanțe:*

- Un proiectant (utilizator) va trebui să poată proiecta sistemul de măști pentru un circuit VLSI dedicat (specializat), cu o densitate de 1 milion/tranzistori per cristal, în circa o lună (circuitul VLSI dorit să poată fi obținut în circa 3 luni).

## 6. Tehnologii de integrare în sistem.

### 6.1. Sisteme inteligente de programare

*Tematica de cercetare.* Elaborarea unui sistem pentru regăsirea de programe dintr-o bancă de algoritmi (bază de cunoștințe) conform unor cerințe ale utilizatorilor de sintetizarea prin inferență a unui program care



să corespundă specificațiilor utilizatorului. Suplimentar, sistemul trebuie să verifice, printr-un proces de inferență, dacă programul generat respectă anumite cerințe de optimalitate.

*Obiective și performanțe:*

- Sistem pentru sinteza și verificarea programelor, bază de programe.

- *Obiectiv intermediar:*

- perfecționări prin sinteza și conversia unor programe pentru domenii specifice, cu o regăsire minimală dintr-o bază de programe;
- elaborarea unei baze de programe de volum mediu;
- generarea unui sistem de verificare funcțională și logică a programelor.

- *Obiectiv final:*

- sinteza de programe complexe, de volum mare, pentru sisteme de gestiune a bazelor de date, procesoare de limbaj etc.;

- elaborarea unei baze de programe de volum mare.

- Sistem de întreținere, dezvoltare și administrare programe.

- *Obiectiv intermediar:* experimente de transformări de echivalențe.

- *Obiectiv final:* un sistem pentru evaluarea performanțelor programelor și perfecționări sistem de transformări de echivalențe.

- Sistem consultant pentru proiectarea programelor:

- *Obiectiv intermediar:* proiectarea de bază.

- *Obiectiv final:*

- întrebare-răspuns în limbaj natural;

- un sistem capabil de a oferi consultanță în proiectarea de sisteme de gestiune a bazelor de date, aplicații cu baze de date etc.

## 6.2. Sisteme de proiectare a bazelor de cunoștințe

*Tematica de cercetare.* Elaborarea unui sistem cu o bază de cunoștințe inclusă organic. Baza de cunoștințe va memora datele tehnice și informațiile necesare proiectării, generării și operării unui sistem de prelucrare a cunoștințelor și va sprijini crearea sistemelor de baze de cunoștințe pornind de la o bază de cunoștințe de bază.

*Obiective și performanțe:*

- Crearea simplă a unui sistem cu bază de cunoștințe care să ofere consultații pentru specialiști ce au nevoie de cunoștințe sofisticate, specializate.

- Baza de cunoștințe proiectată să permită circa 20 000 de reguli.

- Posibilitatea de a putea verifica parțial sistemul la nivelul meta-cunoștințelor. Să fie posibilă o depanare ușoară și în cazul sistemelor cu baze de cunoștințe de volum mare.

- *Obiectiv intermediar:* atingerea a circa 30 % din performanțele finale.



### 6.3. Tehnologii de integrare pentru arhitecturi de sisteme

*Tematica de cercetare.* Tehnici de integrare la nivel de arhitecturi de sisteme de calcul de generația a cincea. Elaborarea de tehnici de construire a sistemelor virtuale și reale, optimizarea configurațiilor și echilibrarea încărcării, proiectarea și elaborarea de sisteme mari, tehnici pentru a asigura fiabilitate ridicată.

### 6.4. Sisteme de baze de date și de baze de date distribuite

*Tematica de cercetare.* Elaborarea unui sistem de baze de date pentru calculatoarele de generația a cincea, tehnici de integrare și utilizare a două sau mai multe baze de date, integrarea sistemelor de baze de cunoștințe.

## 7. Elaborarea bazei tehnologice

### 7.1. Sisteme de dezvoltare

*Tematica de cercetare:* Realizarea în etapele inițiale ale proiectului, a sistemelor suport de dezvoltare: sisteme de proiectare asistată de calculator a circuitelor VLSI; calculatoare personale; rețele de calculatoare; sisteme de elaborare a bazei software și de cunoștințe.

Cele mai importante aspecte ale cercetării-dezvoltării în cadrul proiectului japonez sînt cele legate de arhitecturi și probleme conexe: prelucrare paralelă, elaborarea de software complex eficient, gestiunea eficientă a bazelor de date, sisteme cu funcțiuni distribuite, algoritmi matematici și logici performanți.

Un alt subiect major de cercetare-proiectare în cadrul proiectului japonez este elaborarea unui nou tip de limbaj de programare care să înlocuiască pe cele von Neumann. Orientarea este spre limbajele de tip programare logică și programare funcțională, care vor trebui să ofere inclusiv facilități de programare paralelă și programare fiabilă. Unul din limbajele avute în vedere, ca bază de plecare este limbajul PROLOG extins, care va fi realizat pe bază de inferențe simple cum ar fi *silogismul logic*.

Cu toate că în prima fază a proiectului accentul se pune în mod deosebit pe aspectele de limbaj de programare, în paralel se efectuează și cercetări legate de aspectele de hardware și software de bază. Se lucrează la elaborarea unei mașini microprogramate pentru silogisme secvențiale ce vor servi ca instrumente pentru elaborarea software-ului de prelucrare inferențială paralelă. În domeniul mașinii de bază de cunoștințe, proiectul prevede la început elaborarea unei arhitecturi de mașină de bază de date relațională ce va fi apoi dezvoltată spre o arhitectură finală de mașină de inferențe.

## 11. Etapizarea proiectului japonez

Lansarea propriu-zisă a proiectului japonez a fost precedată de o perioadă de 3 ani (1979—1981) de studii efectuate sub coordonarea unui comitet inter-departamental.



Conform proiectului, realizarea calculatorului de generația a cincea este prevăzută să dureze 10 ani (1982—1991), în trei etape principale (fig. 6) :

— În *etapa 1*, durată 3 ani (1982—1984), se vor elabora elemente tehnologice de bază; se urmărește realizarea unui prototip de mașină tip calculator personal PROLOG (PSI = Personal Sequential Inference Machine) ce va dispune de o bază de cunoștințe comparabilă cu sistemele expert actuale (mii de reguli și mii de obiecte) cu o putere de prelucrare însă de ordinul a 1 milion LIPS, deci cu un ordin de mărime mai mare decât implementările PROLOG prin software actuale (micro-PROLOG pe Z-80 realizează actualmente circa 120 LIPS; interpretoarele PROLOG scrise în limbajul C pentru calculatoarele VAX de 32 biți efectuează circa  $1 \div 3$  KLIPS; Waterloo PROLOG pe calculatoare IBM mari efectuează circa 25 KLIPS, iar compilatorul PROLOG-100 pentru calculatoare DEC 10/20 asigură circa 30 KLIPS, fiind cel mai performant PROLOG actual); acest calculator personal PROLOG japonez va avea și modele comerciale.

— În *etapa 2*, durată 4 ani (1985—1988), se vor elabora subsistemele constitutive, efectuându-se experimente ingineresti precum și experimente de integrare. Obiectivul mai deosebit al acestei etape va fi realizarea *prelucrării paralele*.

— În *etapa 3*, durată 3 ani (1989—1991), se va elabora ansamblul sistemului integrat sub forma unui prototip de supercalculator inferențial, capabil a efectua între 1 milion și 1 miliard LIPS, cu o bază de date ce poate manipula zeci de mii de reguli de inferență și cu sute de mii de obiecte (deci dimensiuni ce corespund memorării, de exemplu, a Enciclopediei Britanice).

Pe baza prototipului funcțional ce se va realiza în cadrul proiectului, diferitele firme constructoare din Japonia vor realiza diferite prototipuri comerciale.

Cel mai interesant și revelator este examinarea etapei 1 din proiectul de cercetare dezvoltare a calculatorului de generația a cincea, fiind o etapă prin natura lucrurilor mai bine definită, care își va pune amprenta hotărâtoare pe celelalte etape (fig. 7).

Temele de cercetare-dezvoltare pentru etapa 1 pot fi sintetizate după cum urmează :

1. *Mașina Inferențială Paralelă* (PIM). Mașina Inferențială Paralelă (Parallel Inferential Machine), împreună cu Mașina Baze de Cunoștințe, formează nucleul hardware-ului calculatorului de generația a cincea. În etapa 1 se va elabora un studiu de evaluare asupra modului de inferență, cuprinzând :

— un mecanism de bază pentru inferențe paralele care să gestioneze executarea în paralel a inferențelor ;

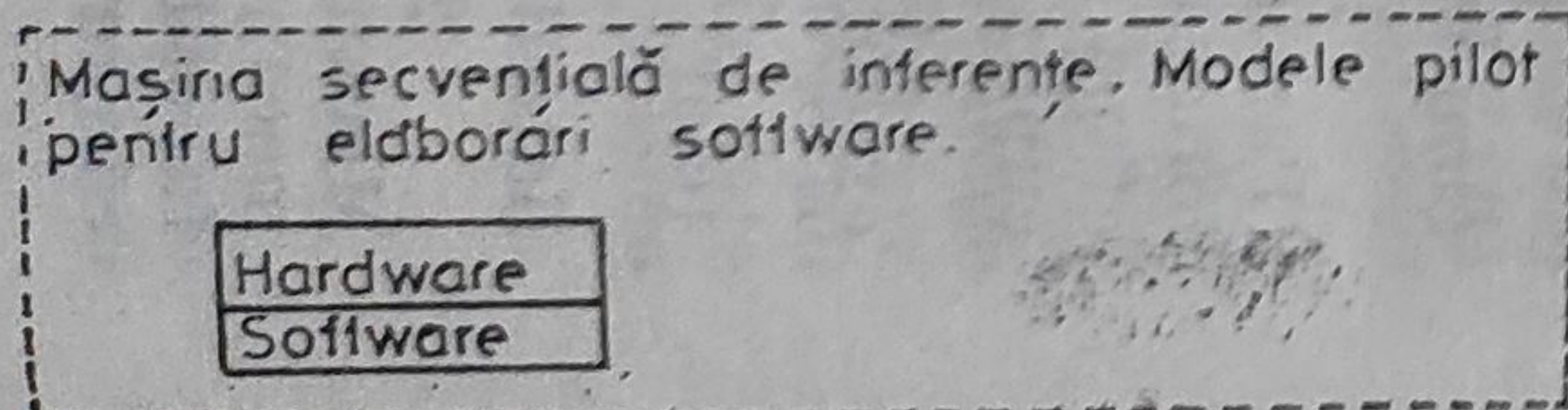
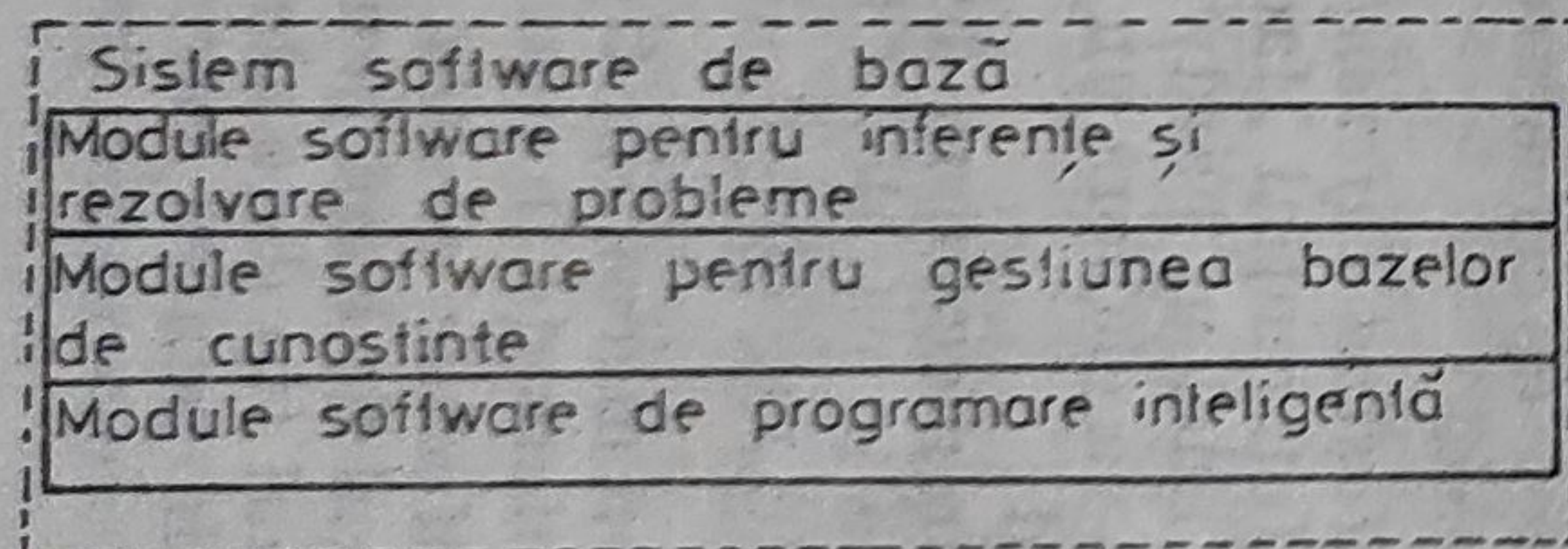
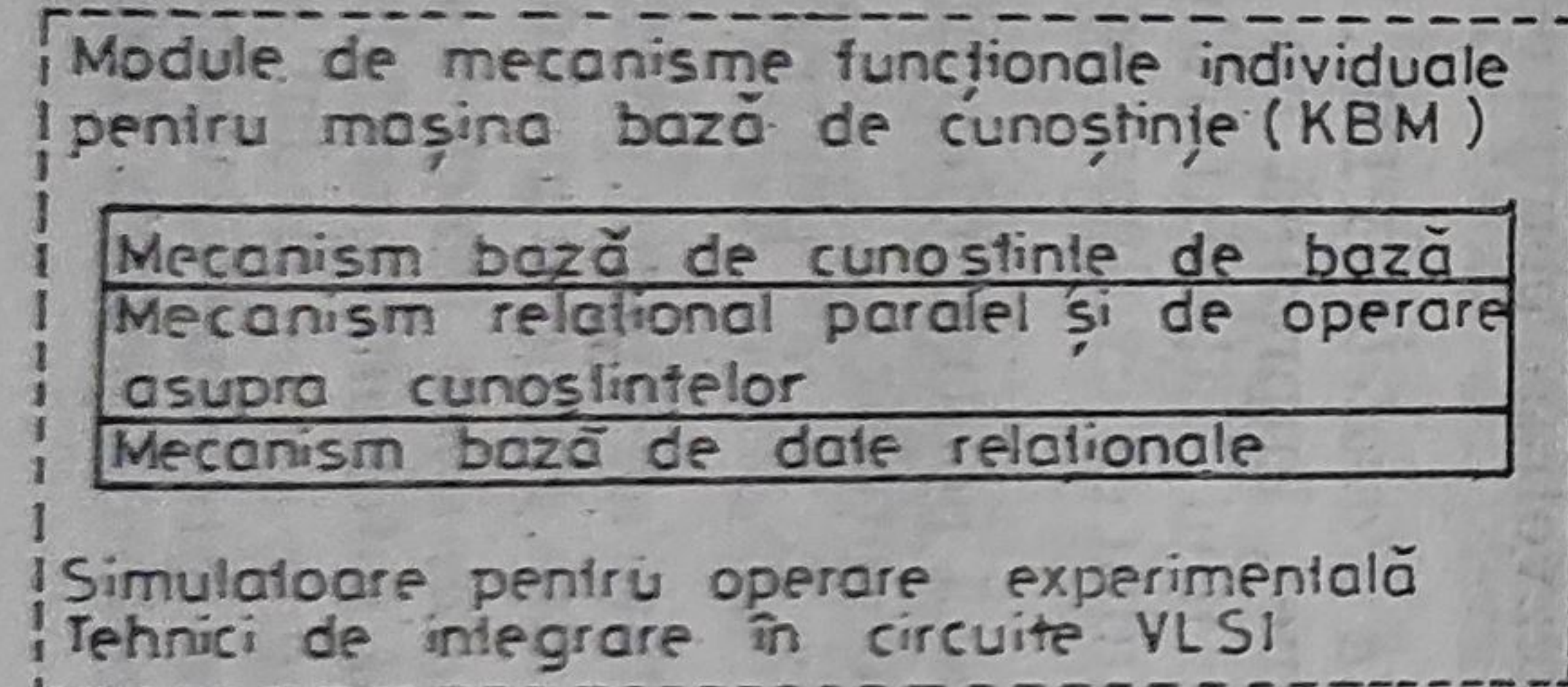
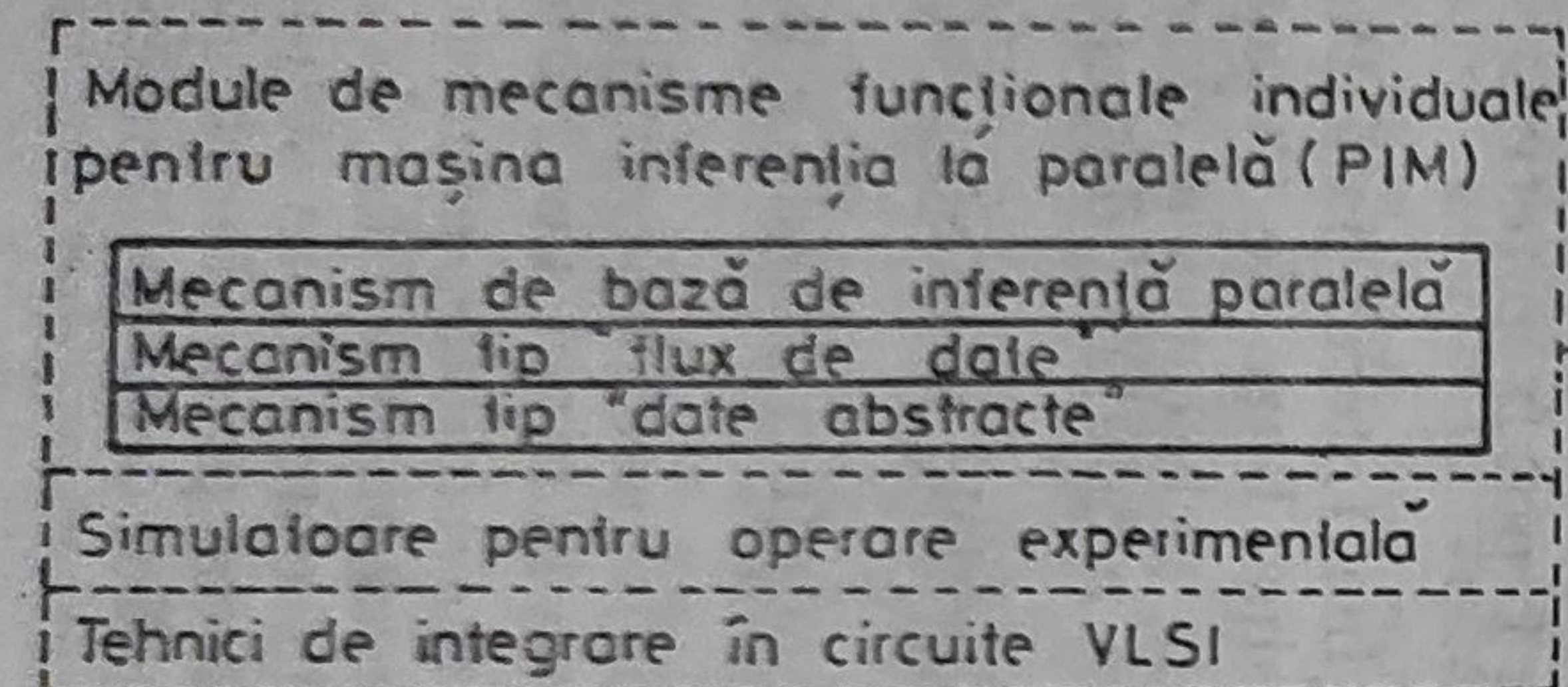
— un mecanism „flux de date” pentru executarea rapidă a inferențelor ;

— un mecanism tip date abstracte pentru agregarea operațiilor detaliate de inferențe în mai multe grupe și gestionarea la nivel de grupe.

2. *Module pentru mecanisme funcționale individuale pentru PIM*. Mecanismul de bază pentru inferențe paralele, mecanismul flux de date și mecanismul tip date abstracte au la rîndul lor în compunere sub-module funcționale pentru care se propune elaborarea de prototipuri, pe baza cărora se vor realiza prototipuri pentru fiecare din cele trei mecanisme.

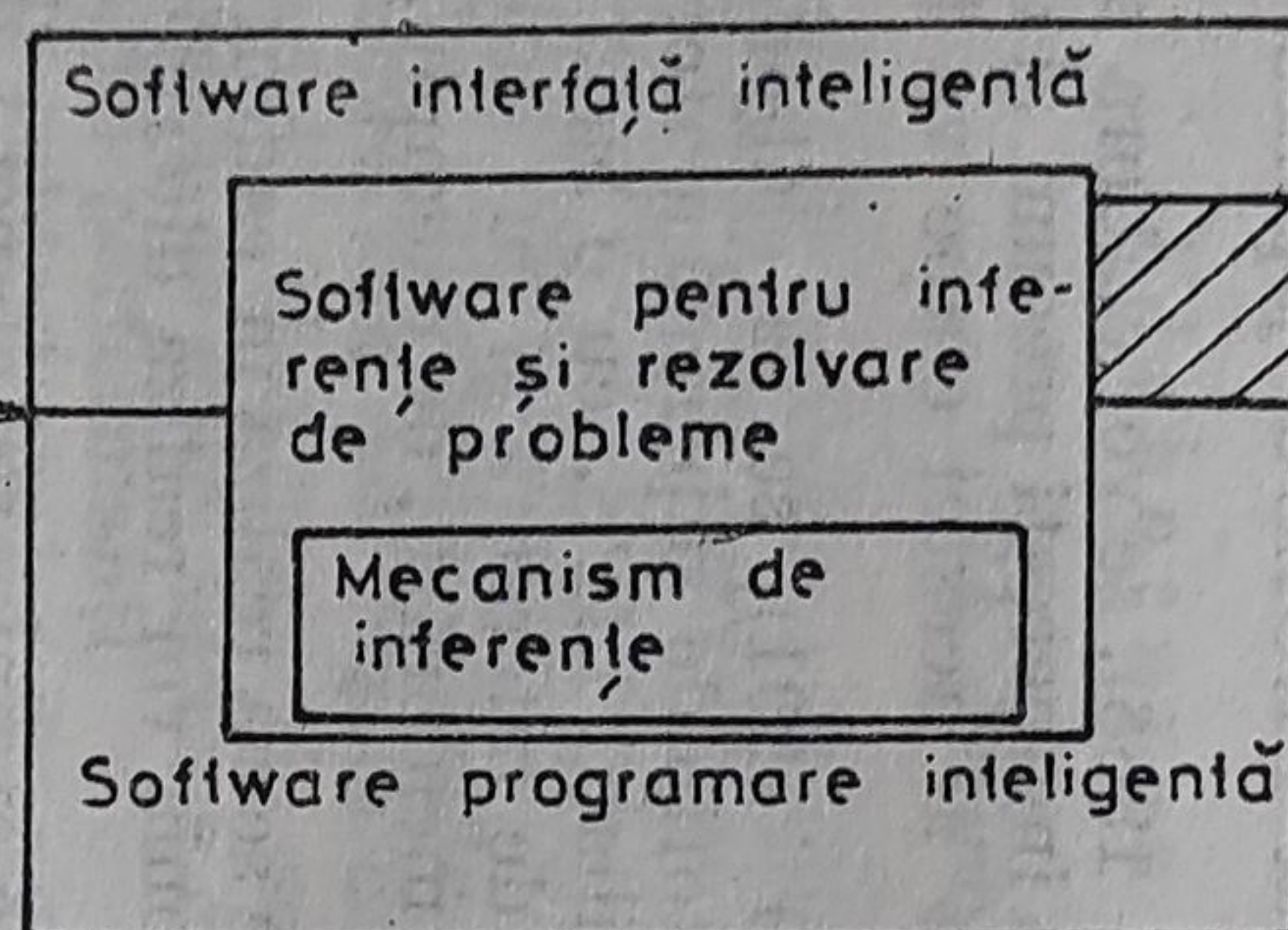


ETAPA ÎNȚIALĂ: ELABORAREA ELEMENTELOR  
TEHNOLOGICE DE BAZĂ

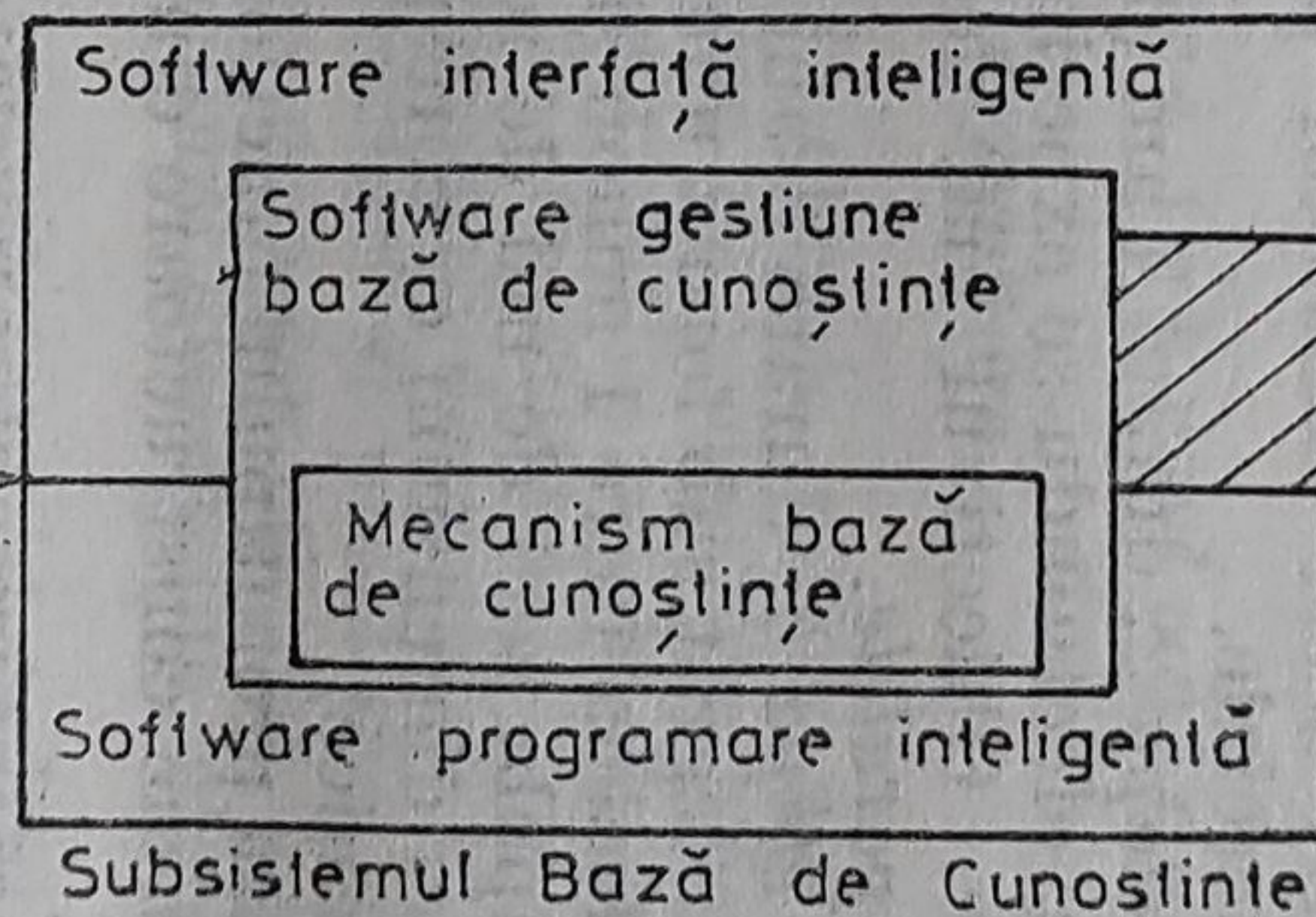


ETAPA INTERMEDIARĂ:  
ELABORAREA SUBSISTEMELOR

Subsistemul Interferențial



Interfața  
inteligentă  
hardware



Interfață  
inteligentă  
hardware

ETAPA FINALĂ:  
ELABORAREA SISTEMULUI  
INTEGRAT

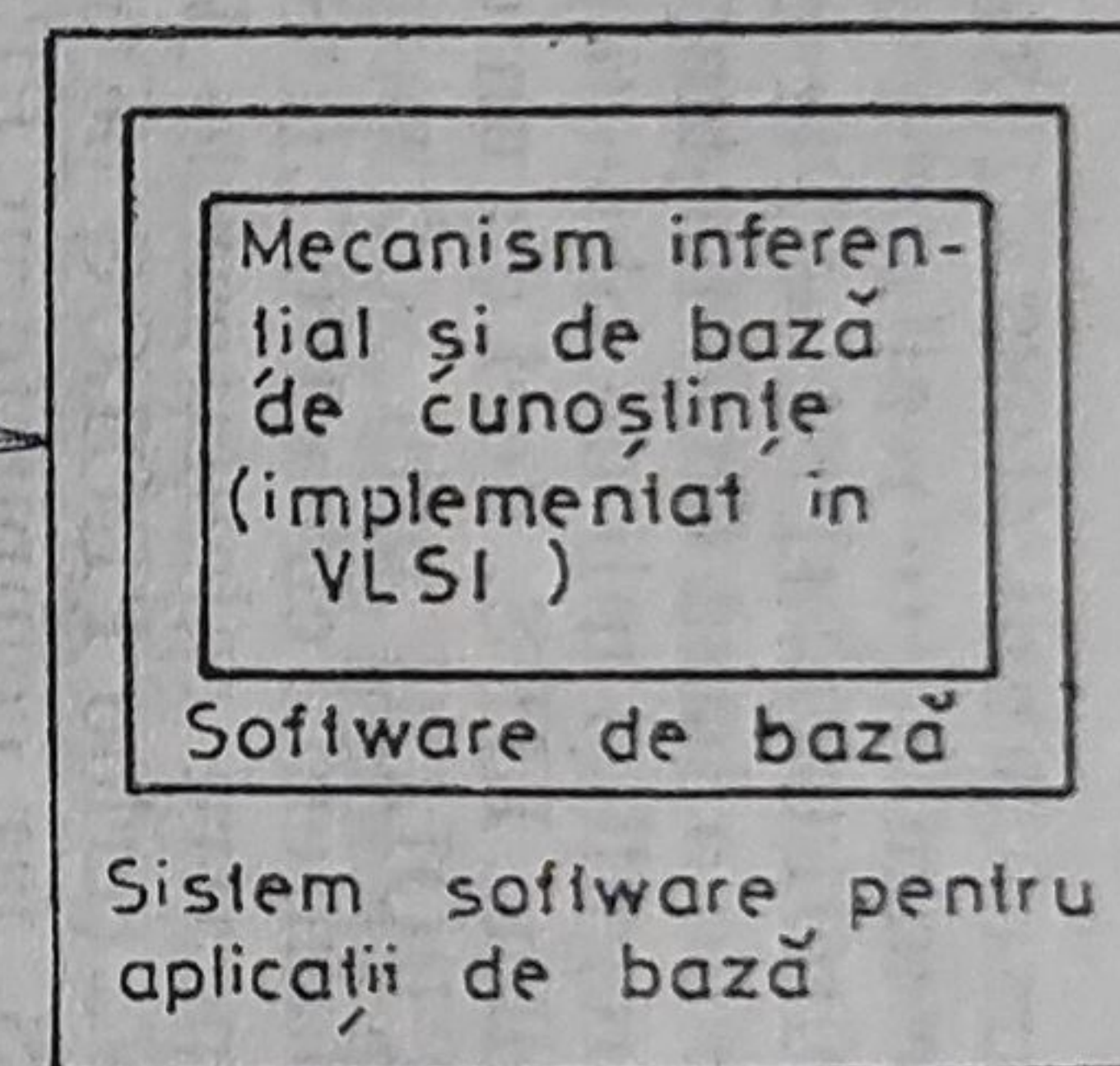


Fig. 6. — Etapele de cercetare-dezvoltare a calculatorului japonez de generația a cincea.



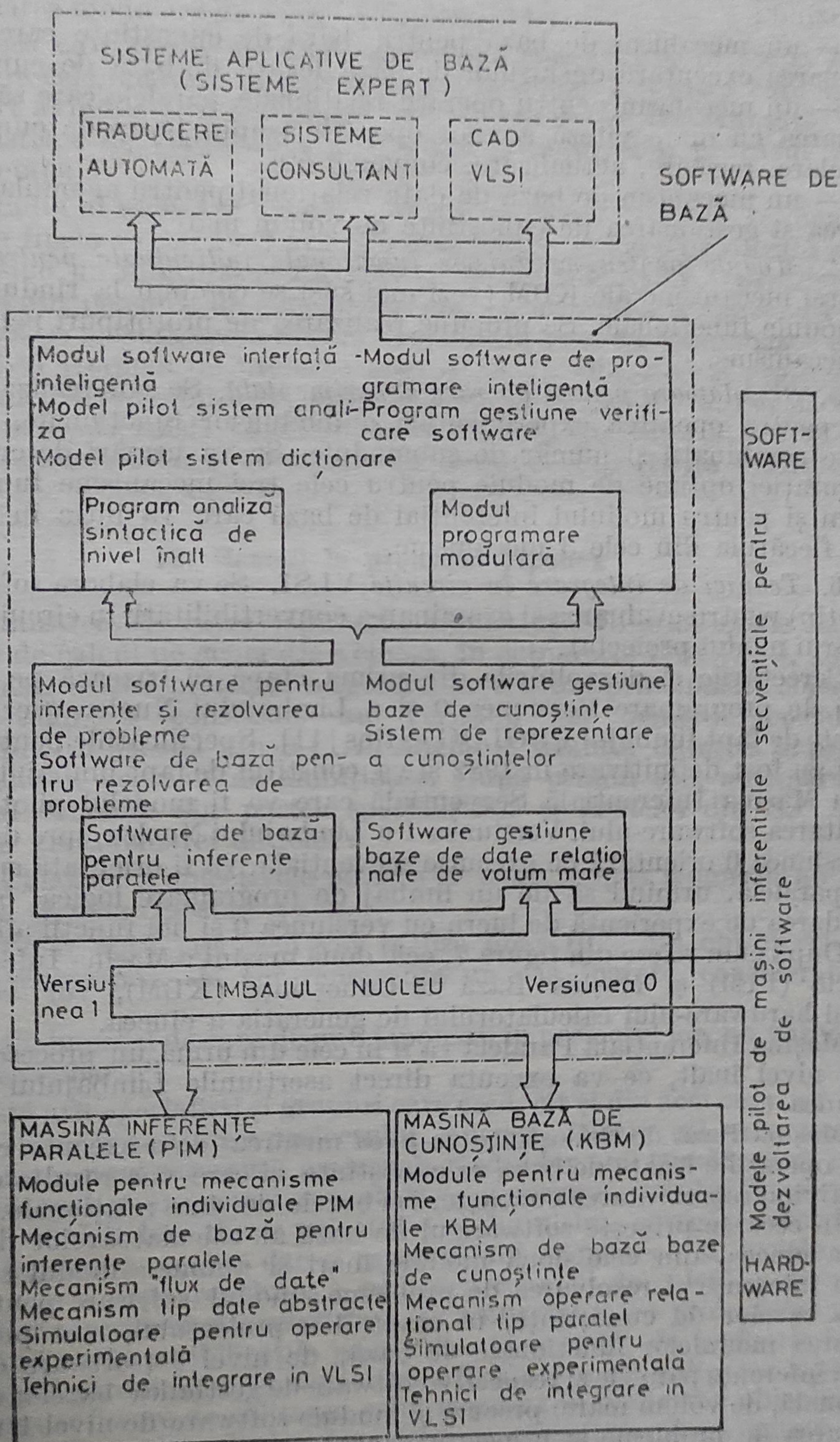


Fig. 7. — Obiectivele de cercetare-dezvoltare pentru etapa 1 a proiectului japonez.



3. *Mașină Bază de Cunoștințe* (KBM). În prima etapă se va elabora un studiu de evaluare a modului de bază pentru baza de cunoștințe, cuprinzând :

- un mecanism de bază pentru baza de cunoștințe care asigură gestionarea executării operațiilor de bază legate de baza de cunoștințe ;
- un mecanism pentru operații relaționale paralele care să asigure efectuarea cu mare viteză a unor operații asupra bazei de cunoștințe : acumulare, regăsire, actualizare, conversii etc.
- un mecanism de bază de date relațional pentru acumularea, memorarea și gestionarea de cunoștințe de volum mare.

4. *Module pentru mecanisme funcționale individuale pentru KBM.* Cele trei mecanisme ale KBM (vezi mai sus) se compun la rândul lor din sub-module funcționale. Se propune realizarea de prototipuri pentru cele trei mecanisme.

5. *Simulatoare pentru operare experimentală.* Se vor realiza simulatoare pentru operarea experimentală a modulelor funcționale, folosind diferite combinații și număr de submodule. Se va urmări determinarea configurației optime de module pentru cele trei mecanisme funcționale precum și pentru modulul inferențial de bază care va intra în compunerea fiecăruia din cele 3 mecanisme.

6. *Tehnici de integrare în circuite VLSI.* Se va elabora software-ul (prototip) pentru evaluarea și examinarea convertibilității în circuite VLSI a fiecărui modul proiectat.

Cercetările și dezvoltările din prima etapă se bazează pe un nou limbaj de programare Versiunea 0 și a Limbajului Nucleu (vezi fig. 7) care este de fapt limbajul PROLOG extins [11]. Specificațiile pentru acest limbaj au fost definitive în 1982 și va constitui de fapt limbajul mașină pentru Mașina Inferențială Secvențială care va fi modelul pilot pentru dezvoltarea software-ului. Versiunea 1 a Limbajului Nucleu, spre deosebire de versiunea 0 orientată pe execuție secvențială, va fi orientată spre execuție paralelă, urmînd să fie un limbaj de programare logică, bazat pe acumularea de experiență de lucru cu versiunea 0 și noi funcții adăugate.

După cum reiese din figura 7, cele două mașini : Mașina Inferențială Paralelă (PIM) și Mașina Bază de Cunoștințe (KBM), vor constitui nucleul hardware-ului calculatorului de generația a cincea.

Mașina Inferențială Paralelă va fi în cele din urmă un procesor paralel de nivel înalt, ce va executa direct aserțiunile Limbajului Nucleu Versiunea 1.

Mașina Bază de Cunoștințe va avea menirea de a executa cu mare viteză operațiile legate de baza de cunoștințe și care vor rezulta din studiile privind reprezentarea cunoașterii și bazele de date relaționale.

În ceea ce privește software-ul de bază al calculatoarelor de generația a cincea, din cele două module mari și anume : modul software pentru inferențe și rezolvarea de probleme ; modul software pentru gestiunea bazelor de cunoștințe, în etapa 1 a proiectului, sînt prevăzute realizarea modulelor funcționale de bază, de nivel 0 (software de bază pentru inferențe paralele și respectiv software de gestiunea bazelor de date relațională, de volum mare) precum și module software de nivel 1 (pentru rezolvarea de probleme și respectiv pentru reprezentarea cunoștințelor).

Pentru software-ul de sistem care se referă la modulul software pentru interfață inteligentă, în etapa 1 se abordează aspectele de analiză semantică și de dicționare (inclusiv analiza sintactică de nivel înalt, ca



aspect complementar), iar pentru modulul software de programare inteligentă, în etapa 1 se abordează problema verificării software-ului (precum și programarea modulară ca aspect complementar).

În ceea ce privește sistemele aplicative elementare, în etapa 1 se au în vedere cele ilustrate în partea de sus a figurii 7, dintre care sistemele expert avînd deja o tehnologie de utilizare destul de avansată, vor servi ca sisteme de verificare a Sistemului Experimental prevăzut a se realiza în etapa 1.

Ca stadiu al realizării obiectivelor din proiectul japonez, se menționează că la finele anului 1983 exista o machetă funcțională realizată software (folosind un interpretor PROLOG), pe un calculator DEC 2060, a unui calculator personal de generația a cincea, urmînd a fi transpusă sub forma finală pe măsură ce va deveni disponibilă baza de componente electronice VLSI (termen de proiect: august 1984). Obiectivele proiectului japonez sînt foarte ambițioase, termenele stabilite inițial sînt considerate (chiar de unii specialiști japonezi) ca optimiste, dar chiar și o depășire a acestor termene cu  $5 \div 10$  ani nu va diminua importanța proiectului.

## 12. Reacții la proiectul japonez

Organizarea de către japonezi a Conferinței Internaționale asupra sistemelor de calcul de generația a cincea, în octombrie 1981 [2, 9, 10], în care au lansat oficial proiectul respectiv, prezentînd principalele obiective ale proiectului, a constituit un veritabil șoc pentru specialiștii din alte țări, în special din S.U.A., Anglia și Franța, datorită în principal obiectivelor deosebit de ambițioase. După primul șoc, au apărut și reacții mai concrete din partea specialiștilor și firmelor din S.U.A. și Europa de vest, remarci de felul:

— „Chiar dacă obiectivele pe care și le-au propus japonezii nu sînt realizabile, trebuie ripostat”; sau

— „Chiar dacă japonezii vor realiza doar 10% din obiectivele pe care și le-au propus, ele vor reprezenta un pas înainte considerabil”; sau

— „Cu bugetul de care dispun japonezii în proiectul lor, nu vor putea ajunge prea departe”.

Există atît specialiști și grupuri care cred, cît și din acei care nu cred în realizabilitatea ca atare a proiectului japonez. Dintre cei care cred în realizabilitatea unui asemenea proiect, la rîndul lor unii consideră că japonezii nu pot realiza un asemenea proiect de anvergură, iar alții îi consideră capabili să ducă la bun sfîrșit proiectul. Cei din ultima categorie afirmă că proiectul japonez are șanse dar și repercusiuni economice și sociale considerabile („o a doua revoluție a utilizării tehnicii de calcul”). Rezerva specialiștilor americani și englezi față de realizabilitatea proiectului japonez se referă nu la posibilitățile japonezilor în materie de inginerie hardware (microelectronică) ci la capacitatea limitată a specialiștilor japonezi de a elabora software de calitate a celui implicat în realizarea obiectivelor de generația a cincea de calculatoare, ceea ce constituie, dacă teama este reală, un handicap foarte important și serios în defavoarea japonezilor.

Invidiat sau denigrat, proiectul japonez rămîne totuși de referință privind generațiile viitoare de calculatoare și a incitat amorsarea unor



proiecte pe subiecte similare, în special în S.U.A., Anglia și alte țări avansate industrial (inclusiv țări socialiste).

Dintre proiectele americane, comparabile ca obiective, mai semnificative sînt 3 sau 4 proiecte. De remarcat că aceste proiecte americane reuneșc interesul și suportul material al mai multor firme constructoare, concurente pe piața calculatoarelor, în unele cazuri susținute și prin credite guvernamentale.

De remarcat și propunerea făcută de americanul Edward Feigenbaum, cunoscut specialist în domeniul inteligenței artificiale, de a crea în S.U.A. un „centru național pentru tehnologii ale cunoașterii”. O părere ce pare că este acceptată de mulți specialiști, economiști, strategii politici și comentatori este că S.U.A. dispune la ora actuală de cei mai numeroși și cei mai buni specialiști în domeniul inteligenței artificiale, iar firmele și agențiile guvernamentale din S.U.A. par că sînt dispuse să investească sume considerabile pentru a traduce această superioritate tehnico-științifică într-o superioritate industrială.

Putem afirma deci că Japonia nu va progresa singură, în mod izolat, spre sistemele inteligente de calcul. Această a doua revoluție a informaticii pe care o reprezintă trecerea de la prelucrarea datelor la prelucrarea cunoștințelor, va fi rezultatul unor eforturi de cercetare-dezvoltare din mai multe țări ale lumii. Meritul Japoniei este de a fi lansat prima un proiect unitar, la nivelul economiei naționale, de coordonare a eforturilor de cercetare-proiectare spre obiective bine definite și structurate. Desigur că un asemenea proiect vast, pe 10 ani înainte, într-un domeniu atît de dinamic ca electronica și tehnica de calcul, este un gest ce poate fi calificat ca temerar și comportă un anumit grad de risc, dar care merită a fi promovat, căci chiar și rezultatele parțiale sînt deosebit de importante.

#### BIBLIOGRAFIE

1. A. E. FEIGENBAUM, PAMELA McCORDUCK, *The Fifth Generation*, Addison — Wesley Publ. Comp., 275 p., 1983.
2. T. MOTO-OKA (editor), *Fifth Generation Computer Systems*, Proceedings of the Internat. Conf. on Fifth Generation Computer Systems, Tokyo, Japan, oct. 19 — 22, 1981, North-Holland Publishing Comp., 1982.
3. MOTO-OKA TOHRU, *Les ordinateurs de cinquième generation*, La Recherche, 134, 516 — 525, april (1984).
4. \* \* \* *The Race to Build A Supercomputer*, Newsweek, 58 — 64, 4 July (1983).
5. PAMELA McCORDUCK, *Introduction to the Fifth Generation*, Communication of the ACM, 26, 9, 629 — 630, sept. (1983).
6. E. Y. SHAPIRO, *The Fifth Generation Project — A Trip Report*, Communication of the ACM, 26, 9, 637 — 641, sept. (1983).
7. R. STEIER (editor), *Cooperation is the key: An Interview with B. R. Inman*, Communication of the ACM, 26, 9, 642 — 645, sept. (1983).
8. P. LEMMONS, *Japan and the Fifth Generation*, BYTE, 304 — 401, nov. (1983).
9. \* \* \* *Fifth Generation Project*, TEHNOCRAT, 15, 1, 29 — 47, ian. (1982).
10. K. FUCHI, *The Direction the FGCS Project Will Take* New Generation Computing, New Generation Computing, 1, 3 — 9, (1983).
11. T. CHIKAYAMA, *ESP — Extended Self — contained PROLOG — as a Preliminary Kernel Language of Fifth Generation Computers*, New Generation Computing, 1, 11 — 24, (1983).
22. I. ILIESCU, *A cincea generație, Inteligența artificială și sfidarea calculatoarelor japoneze adresată lumii*, Note de lector, AMC vol. 44, p. 33, Editura tehnică, 1984.



# CONCEPTE PRIVIND ARHITECTURA SISTEMELOR DE CALCUL DIN GENERAȚIA A CINCEA

EMIL TUDOR\*), LUCIAN NICA\*), MIHAI MĂRȘANU\*)

ARCHITECTURAL CONCEPTS OF THE FIFTH-GENERATION COMPUTERS. The paper starts with a short review of basic conceptual principles involved in the fifth generation computer architecture. Some recent implementations in the frame of fifth generation concepts are then discussed and compared, regarding functional machines (LISP machines), recursive machines, inference and problem solving machines — as well as some dedicated hardware structures for relational data base systems. Finally, a perspective view is achieved on future technological developments in the field of hierarchical high-parallel computer structures.

Prin realizarea proiectului privind generația a cincea de calculatoare, specialiștii japonezi apreciază că Japonia va deține la nivelul anilor 1990 rolul de lider în tehnica de calcul mondială.

Acest proiect [1], menit în concepția autorilor săi să ducă la o a doua revoluție în tehnica de calcul, reprezintă în fapt o continuare la alt nivel a efortului general de dezvoltare a unor sisteme postneumanniene, de perfecționare a structurilor actuale și de depășire a „crizei” software-lui.

Noile arhitecturi prevăzute pentru generația a cincea, se bazează pe dezvoltări substanțiale ale teoriei limbajelor și tehnicii circuitelor VLSI. Ele vor asigura crearea unor sisteme evolute, în dialog direct cu omul (prin folosirea limbajului natural), participând la rezolvarea problemelor prin prelucrarea informației sub forma cunoștințelor.

În configurarea societății viitorului, informaticii îi revine un rol de prim ordin, societatea și sistemele de calcul vor fi într-o strânsă intercondiționare, informația sub forma cunoștințelor devenind o resursă tot atât de importantă și necesară ca energia.

Lucrarea de față își propune să analizeze caracteristicile arhitecturii sistemelor de generația a cincea prefigurate în proiectul amintit și să prezinte unele lucrări de dată recentă care se înscriu pe linia conceptelor acestei arhitecturi.

## 1. Conceptele arhitecturale ale sistemelor de generația a cincea

Proiectul preliminar al generației a cincea propune realizarea unei serii unitare de sisteme de calcul, care vor satisface cerințe funcționale complexe, ca :

---

\*) Institutul de cercetare științifică și inginerie tehnologică pentru tehnică de calcul și informatică.



— un nivel intelectual superior al utilizării sistemelor prin implementarea unor funcții ale comenzii directe (prin voce, imagine, documente scrise) și a funcțiilor de inferență logică prin care se asigură posibilități de prelucrare evoluată a unor mari volume de cunoștințe;

— un cost redus al elaborării programelor asigurat prin limbaje de nivel înalt și arhitecturi corespunzătoare (care să simplifice redarea algoritmilor și corecția programelor) prin perfecționarea ambientului de programe și realizarea unei interfețe inteligente om-mașină;

— un coeficient cost/performanță îmbunătățit, productivități foarte ridicate și memorii de mare volum, care să asigure un nivel înalt de inteligență sistemelor; posibilități multiple de configurare și adaptare la aplicații și un grad înalt de fiabilitate și protecție la accese neautorizate.

Caracteristica esențială a acestor sisteme este orientarea activităților spre prelucrarea informației sub forma cunoștințelor, în condițiile asigurării unei interfețe foarte apropiate de utilizatorul uman.

Cele trei funcții de bază ale sistemelor: rezolvarea problemelor și inferența logică, gestiunea bazelor de cunoștințe și interfața inteligentă, vor fi implementate prin mijloace software și hardware configurate adaptat la specificul aplicației.

Aceste funcții vor caracteriza sistemele la toate nivelurile, de la calculatoarele personale la super-sisteme, implementate la complexități diferite, atingându-se la limita superioară productivității de prelucrare de  $10^8 \div 10^9$  LIPS (inferențe logice pe secundă), volume de cunoștințe de  $10^{11} \div 10^{12}$  octeți și posibilități de obținere a răspunsului în câteva secunde.

O reprezentare sintetică a configurației sistemelor de generația a cincea cuprinde nivelurile de realizare software și hardware a celor trei funcții de bază, interfețele externe și suportul de implementare a sistemului, — arhitecturile VLSI. O astfel de arhitectură de sistem se bazează pe un ansamblu de concepte specifice. Atât descrierea cât și modelarea problemelor se transferă în domeniul interfeței inteligente, activitățile fiind asigurate de limbaje adecvate de nivel înalt. Sistemul înțelege descrierea, corectează eventualele incorectitudini, întocmește modelul și sintetizează programul. Prelucrarea se bazează pe folosirea limbajelor programării logice (tip PROLOG), care permit rezolvarea problemelor prin inferențe logice. Baza de cunoștințe va cuprinde date despre limbaje, sisteme (medii hardware) și aplicații, care vor fi utilizate în procesul analizării datelor de intrare, a modelării și execuției programului, ca și în generarea răspunsurilor. Nivelul foarte înalt al limbajului de programare logică va permite un contact direct om-mașină prin mijlocirea sistemului de modelare software. În sfârșit, în vederea reducerii considerabile a discrepantei semantice actuale manifestate în folosirea sistemelor de calcul, elementele hardware vor prelua o mare parte a funcțiilor software, beneficiind de circuite VLSI evolute.

Sistemul software este structurat pe mai multe niveluri corespunzătoare diferitelor funcții. Nucleul lui este software-ul de bază, cel mai apropiat de hardware, care conține subsisteme pentru fiecare din cele trei funcții de bază. Sistemul inteligent de sistematizare asigură proiectantului funcții inteligente pentru programare, crearea bazei de cunoștințe și a dispozitivelor VLSI inteligente. Sistemul inteligent pentru comandă, exploatare și întreținere, sistemul bazei de cunoștințe și cel al aplicațiilor întregesc complexul software.



În privința arhitecturii hardware se prevăd trei tipuri de echipamente care vor corespunde, de asemenea, funcțiilor de bază. Vor exista astfel, corespunzătoare componentelor actuale ale sistemelor (unitate centrală, memorie ierarhizată, canale și dispozitive de intrare/ieșire), echipamente evaluate pentru rezolvarea problemelor și inferențe logice, pentru baza de cunoștințe și pentru interfața inteligentă.

Arhitectura hardware se va configura din următorul set de structuri hardware, care vor intra în compunerea sistemului în funcție de tipul aplicației și de complexitatea ei: mașina cu programare logică, mașina funcțională, mașina cu algebră relațională, mașina pentru date abstracte, mașina cu flux de date și mașina von Neumann perfecționată.

Interfețele inteligente de limbaj vor utiliza rezultatele elaborării unor noi limbaje din domeniul programării logice și al tratării datelor abstracte. Suportul hardware se va baza pe procesoare VLSI specializate, pentru comunicare directă prin voce, imagine etc.

Sistemele dedicate rezolvării problemelor și inferențelor logice vor include mașina cu flux de date de mare productivitate, care va implementa și metode specifice mașinii funcționale pentru manipularea simbolurilor. Sistemele de mare viteză se vor constitui în jurul unei baze de cunoștințe relaționale (cu o arhitectură care implementează principiile algebrei relaționale), folosind și rezultatele cercetării mașinilor pentru date abstracte. Sistemele de calcul de productivitate relativ redusă vor avea la bază arhitectura von Neumann perfecționată, cu o mare parte a mijloacelor software-lui de bază transferate în hardware.

Toate sistemele vor avea un principiu modular de organizare, iar metodele de interconectare a subsistemelor și tehnica microprogramării vor permite unificarea lor. Implementările vor folosi o gamă largă de circuite VLSI specializate de mare performanță, elaborate cu mijloace ale automatizării proiectării foarte evaluate.

Sistemele de calcul de generația a cincea de diferite specializări, calculatoare personale, sisteme pentru rezolvarea problemelor și inferențe logice, sisteme dezvoltate pentru bazele de cunoștințe, se vor interconecta în rețele locale și globale, utilizând mijloace de transmitere a datelor foarte rapide. Acestea vor avea un limbaj comun de programare, constituindu-se în familii, chiar dacă cuprind sisteme dedicate.

În continuare sînt prezentate unele din soluțiile actuale adoptate pentru arhitecturile evaluate, care ar putea constitui o bază de plecare pentru dezvoltarea sistemelor din generația a cincea.

## 2. Mașini funcționale, prelucrarea simbolurilor

Se recunoaște unanim că la baza „crizei” software-lui stă nivelul scăzut al arhitecturii hardware care a făcut necesară o dezvoltare anormală a mediului software, obligat să se apropie de nivelul de abstractizare al utilizatorului. Diferența dintre posibilitățile intelectuale umane și mediul hardware disponibil constituie „discrepanța semantică” (semantic gap) și o cale de reducere a acesteia este dezvoltarea arhitecturilor de înalt nivel [2]. Progresul considerabil al sistemelor VLSI asigură o perspectivă acestui deziderat.



Mașina funcțională prevăzută în proiectul japonez va avea o arhitectură adecvată prelucrării simbolurilor, va accepta limbaje tip LISP și va folosi modalitățile de calcul paralel. Cercetările vizează dezvoltarea unor structuri paralele, în flux de date și asociative, a unor mașini LISP individuale, a mașinilor gazdă universale cu dispozitive VLSI pentru manipularea simbolurilor etc.

Trebuie remarcat că preocupările pentru elaborarea noilor arhitecturi HLL datează încă din deceniul al 7-lea, dar ele nu au dus la soluții comerciale (exceptând sistemele Burroughs).

Dezvoltarea acestor mașini a fost frânată de existența unor mituri asociate arhitecturii HLL [3]: mașina trebuie dedicată unui singur limbaj, compilatoarele hard sînt simple și ieftine, compactarea codului este un câștig major etc., care trebuie reconsiderate. Soluțiile preconizate ca viabile trebuie să confirme arhitecturi de uz general capabile să suporte limbaje diferite (eventual prin microprogramare dinamică), arhitecturi adaptabile pe aplicații etc. Aceste soluții vor putea deveni comerciale dacă vor oferi o creștere a productivității cu un ordin de mărime.

Un șir de realizări relativ recente (1979—1982) încearcă satisfacerea acestor deziderate. La Mitsubishi s-a cercetat un compilator VLSI PASCAL [4], la MIT s-a realizat un microprocesor LISP denumit SCHEME-79 [5], iar un grup de cercetători de la Universitatea din Toulouse au realizat la comanda INRIA o arhitectură LISP [6]. În Japonia există, de asemenea, un interes manifestat de companii ca Fujitsu și NTT pentru construirea unor mașini LISP de mare productivitate [7].

În viziunea cercetătorilor de la Mitsubishi, dispunerea unui compilator pe cipuri VLSI permite dotarea sistemelor cu un set de compilatoare VLSI care vor executa compilarea în cadrul unor terminale inteligente, degrevînd sistemul de această sarcină (în spiritul conceptelor proiectului pentru generația a cincea). Compilatorul PASCAL este format din patru dispozitive (pentru analiza lexicală, sintactică și pentru generarea codului); dispozitivul pentru analiza sintactică va fi realizat cu circuite PLA pe un cip de  $6 \times 8$  mm.

Pentru implementarea SCHEME-79 s-a utilizat un limbaj asemănător cu LISP. Programele LISP sînt transformate în programe mașină, păstrîndu-se structura expresiilor ce vor fi analizate de evaluator. Programul obiect este structurat ca o listă de pointeri caracteristici, fiecare conținînd un cîmp de date și un cîmp cu tipul expresiei, care este analizat sintactic pentru a se determina operația de executat. Aceasta poate fi furnizarea unei valori imediate, o condiție de evaluare a unei alternative de două subexpresii sau evaluarea unei proceduri pentru argumente date. Parcurgerea arborelui expresiei este recursivă.

Setul de registre cu care lucrează evaluatorul cuprinde registre pentru valoarea curentă a expresiei, pentru menținerea valorilor argumentelor necesare procedurilor, pentru revenirea în program după evaluare etc. Microcodul de comandă este de două tipuri: vertical și orizontal (pentru a reduce volumul microcodului la redarea instrucțiunilor identice ca operații). Interpretorul este scris în LISP (cu unele primitive de adaptare la structura hard) și prin compilări succesive a rezultat microcodul de control. Implementarea VLSI s-a făcut cu ajutorul unui limbaj de amplasare, bazat tot pe LISP.

O realizare interesantă este cea a grupului de la Toulouse, care și-a concentrat eforturile spre elaborarea unei structuri capabile să interpreteze



direct limbaje de tip LISP cu performanțe superioare și folosind tehnica microprocesoarelor. Mașina M3L (de la LISP-Like-Language), operațională din 1980, se compune din două procesoare, un editor pentru traducerea în forma 3L și un interpretor care interpretează în hardware forma 3L (vezi fig. 1). Un limbaj înalt de microprogramare (LEM) specializat în manipularea eficientă a elementelor specifice LISP (celulele cu doi pointeri și un descriptor) este în același timp limbajul de microprogramare pentru procesorul M3L.

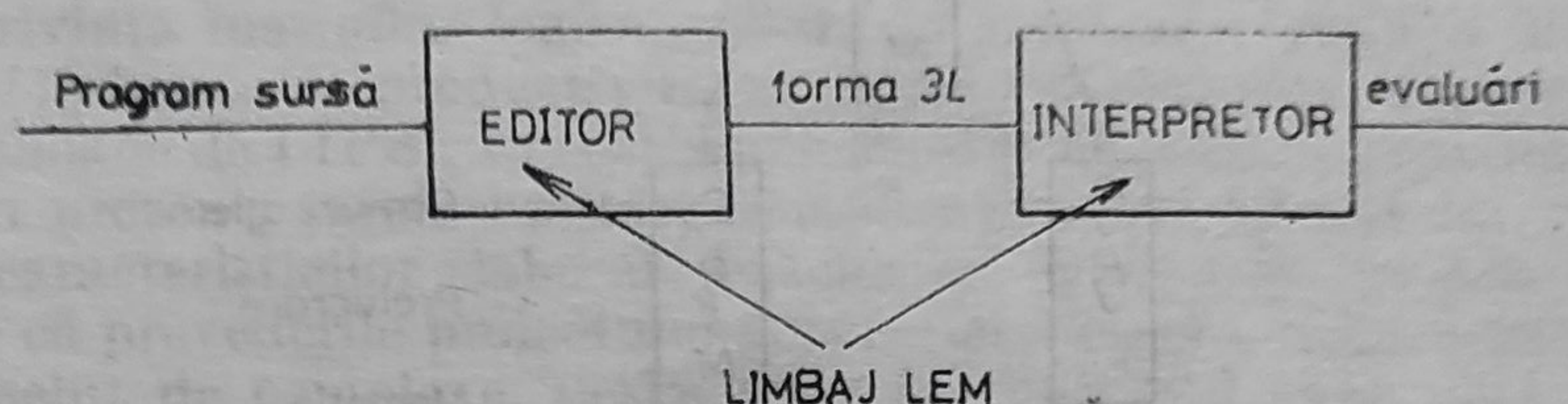


Fig. 1. — Mașina M3L.

Compilarea procedurilor LISP duce la coduri de microprogram foarte concise datorită apropierii semantice dintre LEM și LISP. Susținerea funcțiilor efective de prelucrare, diferite de prelucrarea formală, cele aritmetice, de intrare/ieșire etc., a fost realizată prin microprocesorul AMD2900. S-a ajuns la un cod foarte compact pentru interpretor (circa 1 300 microinstrucțiuni) și la obținerea unei viteze de  $5 \div 15$  ori mai mari față de implementările software pe PDP (față de cipul MIT-SCHEME 79 viteza este de  $5 \div 11$  ori mai mare și sistemul este autonom).

### 3. Arhitectura recursivă

De la definirea principiilor mașini recursive, de către Gluşkov în 1974, au fost realizate multe implementări conforme acestui tip arhitectural. Pe linia cercetărilor sugerate de proiectul generației a cincea privind prelucrarea în flux de date într-un sistem VLSI multi-microprocesor, se înscriu și lucrările de la Universitatea din Newcastle (Anglia), care propun o arhitectură recursivă bazată pe un set de microprocesoare identice (vezi fig. 2), fiecare dotat cu facilități de prelucrare, memorare și comunicații [8]. Această structură ne-neumanniană tinde să satisfacă următoarele principii, care ar defini mașina recursivă: ierarhie de celule de memorie de lungime variabilă delimitate ca șiruri, spațiu de adresare contextual, limbaj mașină recursiv, comandă paralelă descentralizată și organizare în rețea a elementelor de calcul. Informația este reprezentată în șiruri întrepătrunse (nested) de lungime variabilă, formate din date și delimitatori. Adresa este contextuală, fiind redată de o secvență de selectori care indică traseul pe arborele de informație de la un punct de referință la șirul destinație. Programul este redat de instrucțiuni cu format unic, cu sintaxă recursivă, care constau dintr-un șir de argumente, unele putând fi la rândul lor instrucțiuni.

Execuția instrucțiunii este similară cu cea a unei proceduri apelate prin nume; operația (primul argument al șirului) definește un procesor virtual care operează asupra argumentelor-operanți, fiind posibile evaluări paralele. Procesoare cu rol de procesoare virtuale sînt alocate la cerere



în momentul execuției și evaluează în paralel argumentele. Această structură recursivă a fost experimentată pe o implementare simplă realizată cu cipuri VLSI, urmărindu-se în principal verificarea principiului și nu obținerea unor performanțe înalte.

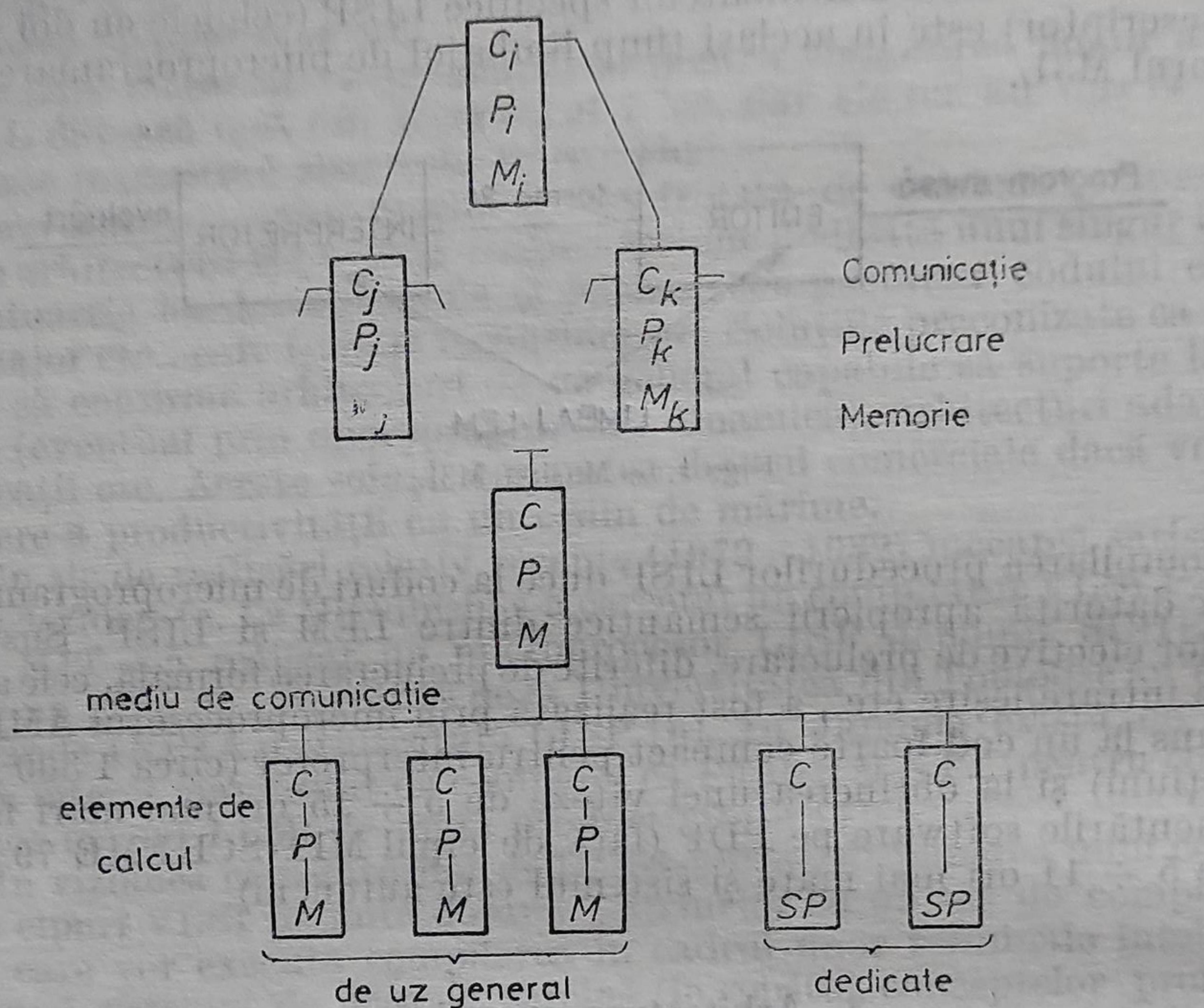


Fig. 2. — Arhitectura recursivă.

#### 4. Mașini de inferențe logice

În cadrul proiectului generației a cincea s-a indicat limbajul PROLOG ca limbaj mașină pentru sistemele de rezolvare a problemelor și inferențe logice, care vor asigura productivități cu un ordin de mărime superior sistemelor software actuale care implementează PROLOG-ul.

Elaborat în jurul anului 1970 la Universitatea din Marsilia (de Alain Colmerauer), Prolog-ul este un limbaj de programare logică destinat prelucrării simbolice a datelor. Programele și datele sînt redată într-o structură uniformă. Specifică pentru acest limbaj este posibilitatea de descriere logică a unor obiecte și a relațiilor dintre ele, sistemul căutînd să deducă soluții prin inferență logică. Programatorul nu indică ce operații trebuie să se execute și cînd, ci specifică un set de ipoteze și solicită o soluție validă. Aceasta duce la ușurință în citirea programelor, la conciziune și claritate. Controlul efectuării procesului de calcul este specificat atît prin semantica declarativă logică a limbajului, cît și prin informația explicită dată de programator, dar el depinde și de inferențele logice efectuate pe parcurs [9]. Aceste caracteristici, care au dus la o utilizare relativ largă a limbajului Prolog în multe domenii ale inteligenței artificiale, cum ar fi bazele de date relaționale, logica matematică și înțelegerea limbajului natural etc., pot justifica alegerea lui ca limbaj mașină pentru sis-



temele generației a cincea. În plus, Prolog-ul, ca limbaj logic, este adecvat prelucrării paralele, larg utilizate în sistemele generației a cincea. Totuși această alegere este controversată; se reproșează limbajelor logice incapacitatea de susținere a buclelor adânc întrepătrunse, ca și lipsa unor facilități specifice pentru manipularea expresiilor aritmetice și rezolvarea ecuațiilor; unele reprezentări sînt dificile de înțeles, iar anumite detalii ale rezolvării problemelor devin invizibile pentru utilizator, ceea ce poate conduce la activități de căutare a soluțiilor extrem de laborioase.

În privința mașinilor logice există, de asemenea, rețineri ale unor specialiști [7] legate de productivitatea foarte ridicată care se prevede în proiect (milioane de LIPS). În susținerea acestor obiecții se argumentează, că, pînă în prezent, rarele limitări în aplicații s-au datorat nu vitezei reduse ci caracteristicilor slabe ale bazelor de cunoștințe. Se pierde însă din vedere că prevederile proiectului generației a cincea vizează prelucrări logice deosebit de complexe, desfășurate în contextul unor baze de cunoștințe vaste.

La sfîrșitul anului 1983 s-a anunțat realizarea la ICOT (împreună cu Mitsubishi El.Co.) a primei mașini inferențiale de uz personal, care execută limbajul-nucleu „0” (din aceeași familie cu Prolog) și va fi utilizată la dezvoltarea software-lui pentru generația a cincea. Performanțele sînt remarcabile: memorie instalată de 20 Mocteti, memorie externă pe două discuri de cîte 38 Mocteti, viteza de 30 KLIPS (care egalează implementarea Prolog pe calculatorul DEC) [10].

## 5. Mașini pentru baze de date relaționale

În proiectul japonez se prevăd cercetări pentru elaborarea mașinilor capabile să folosească algebra relațională (nucleul viitoarelor baze de date) ca limbaj de interfață. Cercetările vor include elaborarea limbajului de interfață, dezvoltarea sistemului de comandă al bazei de date și a elementelor procesoare și de memorie, elaborarea arhitecturii mașinii etc. Lucrările existente privind bazele de date (inclusiv cele relaționale) constituie suportul acestei dezvoltări.

La peste 10 ani de la fundamentarea de către E. F. Codd de la IBM a modelului bazelor de date relaționale, o serie de realizări confirmă avantajele acestei organizări bazate pe structurarea datelor sub forma tabelelor (ordonate pe linii și coloane, unde liniile reprezintă înregistrările, iar coloanele, cîmpuri ale acestora). Între aceste avantaje se menționează ușurința înțelegerii structurii datelor de către utilizator, o independență mai mare a datelor (față de structurile ierarhice sau în rețea), posibilitatea unei baze teoretice pentru operațiile cu tabele, precum și definirea datelor prin limbajul acestor operații [11].

Limbajele relaționale algebrice (de tip SEQUEL, SQL, ale IBM) oferă posibilitatea specificării explicite a operațiilor cu tabele și includ funcții care operează asupra înregistrărilor, similare celor din teoria mulțimilor (uniune, intersecție etc.). Alte limbaje (QBE, OBE, ale IBM) sînt special destinate lucrului cu terminal tip display și permit compunerea tabelelor (relații) conform unor modele, ca și a interogărilor. Limbajul QBE (QUERY-by-Example) s-a implementat într-un sistem comercial (IBM), apreciat pentru ușurința deosebită în utilizare [13].



Sistemul R al IBM se bazează pe limbajul relațional algebric SQL folosit interactiv ca sublimbaj de date în cadrul programelor scrise în PL/1 sau COBOL. Se evită astfel scrierea dificilă de către utilizator a programelor de tranzații în aceste limbaje [12]. Un preprocesor tratează instrucțiunile SQL în cadrul programului, translatându-le în apelări la proceduri specializate ale sistemului R.

Eficiența modelelor relaționale este redusă deoarece atât modelul cit și limbajul de nivel înalt sînt interpretate la nivelul scăzut al hardware-ului. De aceea, în afară de sistemele software implementate pe sisteme de calcul de uz general s-au dezvoltat sisteme care utilizează un hardware dedicat pentru comanda bazelor de date. Un calculator „gazdă” (front-end) compilează instrucțiunile de nivel înalt ale limbajului relațional într-un set de instrucțiuni mașină și un set de procesoare preiau sarcina execuției eficiente a primitivelor specifice managementului bazelor de date. Acestea cuprind operații de căutare a coincidenței de modele (pattern matching), de modificare a informației de control și de actualizare de date. Între primele elaborări se evidențiază sistemele RAP ( Relational Associative Processor) [14] și CASSM (Context Addressed Segment Sequential Memory) [15]. Pentru creșterea eficienței, aceste sisteme folosesc conceptul „logică asociată pistei” (logic-per-track) în care fiecărui volum de date i se atașează logica specifică de prelucrare; în acest fel nu se mai consumă timp și resurse cu transferul informației între două niveluri de memorie (principală și secundară). O altă caracteristică a acestor mașini o constituie prelucrarea paralelă, care decurge din paralelismul explorării segmentelor pe disc, înregistrate pe diferite piste ale unui cilindru.

Arhitectura generală de tip celular a CASSM este redată în figura 3.

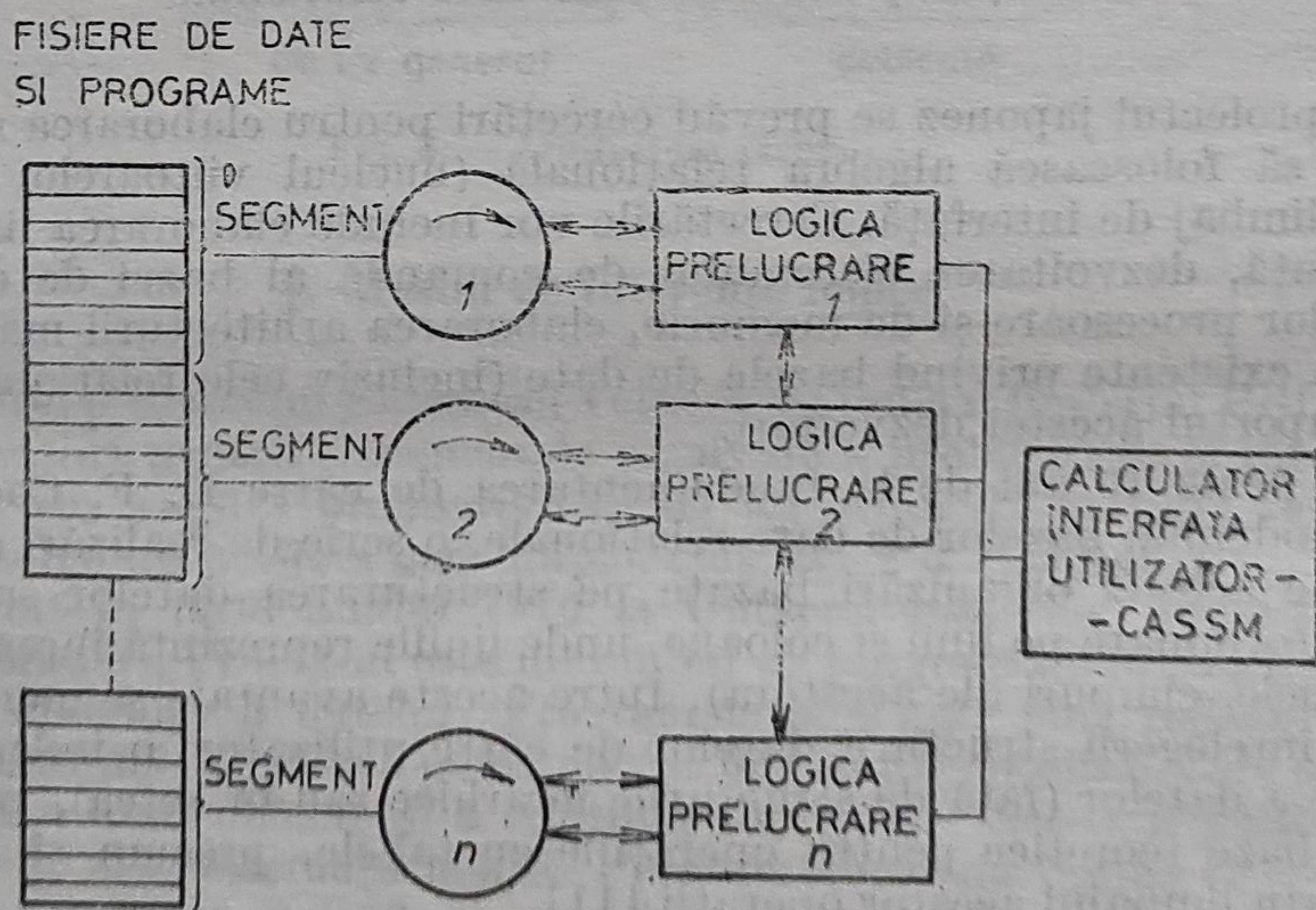


Fig. 3. — Arhitectura sistemului CASSM.

Calculatorul de interfață suportă trecerea de la nivelul înalt al limbajului utilizator la hardware-ul specific bazei de date, compus dintr-un set de procesoare, fiecare prelucrînd logic informația citită (înscrisă) a unui element de memorie rotativ. Atît datele cit și programele compilate sînt conținute în elementele de memorie asociativă, sînt preluate și prelucrate de procesoare. Fiecare element de prelucrare este compus dintr-un



set de dispozitive care asigură prelucrarea pipeline orientată spre operații nenumerice, de căutare în structuri de date complexe, având implementate și tehnicile convenționale (controlul buclor, apelarea subrutinelor etc.). Specifică deci pentru CASSM este prelucrarea paralelă a programelor de către procesoare independente de calculatorul de interfață (sisteme ca RAP în care programele se execută într-un controler pot fi afectate de limitările de memorie ale acestuia). Se consideră că nivelul costurilor unui astfel de sistem nu este prea ridicat, întrucât folosește celule cu structură identică, iar memoria poate fi de tip CCD sau cu bule.

Pentru creșterea eficienței de prelucrare s-au elaborat structuri multiprocesoare capabile să prelucereze simultan mai multe fluxuri de instrucțiuni. Sistemul DIRECT [16] este configurat cu elemente PDP și efectuează execuția simultană a unor interogări relaționale de la diferiți utilizatori, concomitent cu execuția în paralel a fiecărei interogări.

Cercetătorii japonezi de la Universitatea din Yokohama au implementat în hardware un sortator pentru o bază de date cu un debit de transfer de 3 Mocteți/s, organizat ca un arbore binar în care fiecare nod poate fi realizat cu un cip LSI. Funcțiile primitive pentru manipularea datelor se execută printr-un hardware suplimentar, ceea ce reduce volumul software-lui și permite utilizatorului o viziune de nivel înalt [17].

Utilizate cu succes mai ales în aplicațiile în care datele nu au o structură ierarhică implicită, bazele de date relaționale au o deosebită dezvoltare în domeniul prelucrării imaginilor, existând deja la nivelul anului 1981 cel puțin 10 realizări în domeniu (majoritatea în universități, două din ele în Japonia) [18].

## 6. Perspectiva implementărilor tehnologice pentru structuri paralele și ierarhizate

Programul american [19] pentru realizarea circuitelor integrate de foarte mare viteză (VHSIC) prevede următoarele obiective, eșalonate după evoluția factorului de merit (definit ca produsul dintre numărul de porți și frecvența ceasului) și anume : pe termen scurt, realizarea în tehnologia de  $1,25 \mu$  de circuite cu factor de merit de minim  $5 \times 10^{11}$  porți  $\times$  Hz/cm<sup>2</sup> (ceasul de 25 MHz și puterea disipată maximă 3 W/cm<sup>2</sup>); pe termen lung, realizarea în tehnologia de  $0,5 \div 0,8 \mu$  de circuite cu un factor de merit de minim  $10^{13}$  porți  $\times$  Hz/cm<sup>2</sup>.

Tehnologia actuală, datorită restricțiilor privind dimensiunea elementelor și puterea disipată, nu permite implementarea tuturor funcțiilor unui calculator de mare viteză într-un singur cip și atunci s-a propus partiționarea pe principalele 5 funcțiuni/cipuri : unitatea centrală (CPU), unitatea de management a memoriei (MMU), controlerul I/O (IOC), memoria și unitățile de interfață I/O.

În [19] se propune o structură cu 3 busuri, pentru procesor, memorie și intrare/ieșire, MMU și IOC având și un rol de tampon între busul procesorului și celelalte două busuri.

Busul procesorului este un bus protejat, cu o capacitate limitată de comandă a sarcinilor pentru a limita disipația, fizic este foarte scurt și poate fi amplasat pe o singură placă imprimată împreună cu CPU, MMU și IOC.



O astfel de structură compactă implementată cu tehnologia actuală, pentru aviație, poate asigura deja o productivitate maximă de 6 MIPS (pentru o memorie cu timp de acces de 50 ns, cache de 256 cuvinte de 40 de biți, multiplicator hardware, ROM cu 2 Kcuvinte, microinstrucțiuni de 80 de biți cu timp de acces de 15 ns).

Un alt cunoscut proiect american denumit INFOPLEX [20] vizează realizarea unui sistem multiprocesor cu un subsistem ierarhizat de memorare ca suport optim pentru baze de date foarte mari. Sistemul este de tip  $N$  procesoare și  $M$  memorii cache și are 6 niveluri de memorare (vezi fig. 4).

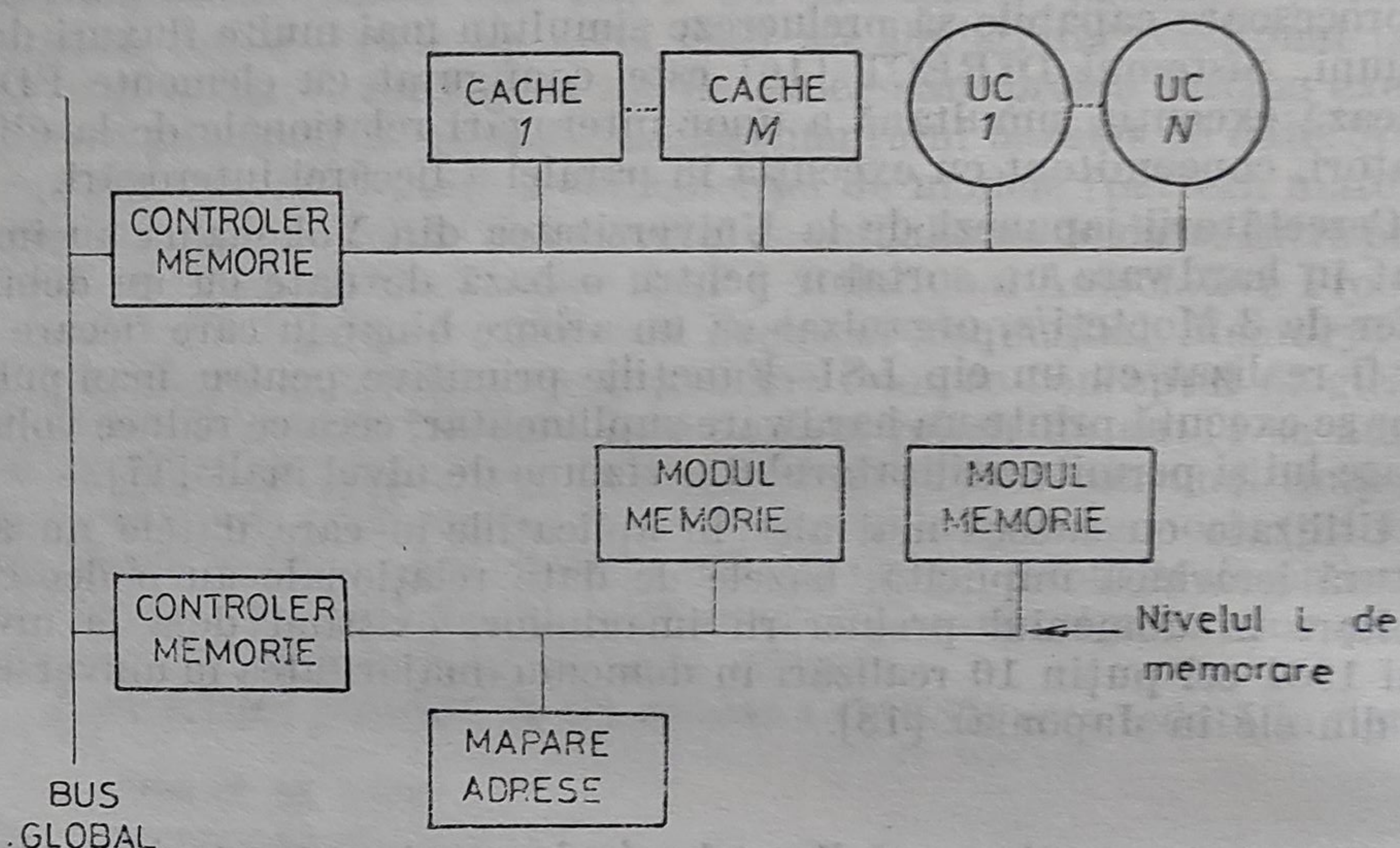


Fig. 4. — Sistemul INFOPLEX.

Nivelul nr. 1 al celor  $M$  memorii cache este utilizat în comun printr-un bus local, de către toate cele  $N$  procesoare, fiind rezolvată problema consistenței copiilor în memoriile cache. Comunicația între diverse niveluri (cu busuri locale și unități de memorare) se realizează printr-un bus global.

Cele șase niveluri ierarhice de memorare includ o gamă largă de tehnologii care coexistă în 1985 și sînt distribuite pe o plajă de valori de 7 ordine de mărime pentru timpul de acces și 5 ordine de mărime pentru costul per octet: memorii cache ultrarapide (20 ns), memoria internă rapidă (200 ns), memoria cu circuite bubble sau CCD, tamburul magnetic, unitățile de discuri magnetice și memoria arhivă tip IBM 3850.

Se estimează că o astfel de structură, pentru un flux de cereri în care majoritatea operațiilor reprezintă citiri de date și beneficiind de tehnologiile anului 1985, poate suporta un debit de câteva milioane de cereri/s cu un timp mediu de răspuns de ordinul câtorva  $\mu$ s.

În contextul acestui deceniu, cînd progresele așteptate în domeniul memoriilor externe sînt mai lente decît cele din domeniul procesoarelor VLSI, proiectul INFOPLEX ilustrează un mod global de abordare a problemei creșterii productivității sistemelor de calcul. O analiză a modelării performanțelor memoriilor externe rotaționale se dă în [21] și [22].



În lucrarea [23] se prezintă conceptele pentru creșterea în continuare a productivității sistemelor de calcul cu discuri magnetice, vizînd realizarea unor tipuri noi de dispozitive funcționale, de optimizare a timpului de acces la date pe suporturi magnetice rotaționale, dispozitivele fiind proiectate cu structuri electronice actuale și în perspectivă cu structuri neconvenționale.

Idealizarea paralelismului activităților în sistemele de calcul a condus la conceptul de paracomputer, care constă dintr-un mare număr de procesoare autonome ce au acces la o memorie centrală, un număr oricît de mare de citiri și scrieri simultane la aceeași celulă de memorie fiind efectuate într-un singur ciclu. O asemenea structură ideală, destinată aplicațiilor științifice de mare complexitate computațională, nu poate fi construită practic. Pe plan mondial au loc însă mai multe încercări de a aproxima structura ideală. Dintre acestea, ultracalculatorul NYU reprezintă o aproximare a unei astfel de structuri realizabilă pînă în anul 1990, în care s-a înlocuit accesul direct într-un ciclu la memoria partajată, printr-un acces indirect multiciclu via rețeaua de conectare.

Proiectul NYU [24] prevede realizarea unui sistem cu 4 096 de procesoare (în tehnologia anului 1990), cu acces la o memorie centrală partajată, prin intermediul unei rețele de comutare de mesaje cu o geometrie de tip rețea OMEGA (vezi fig. 5).

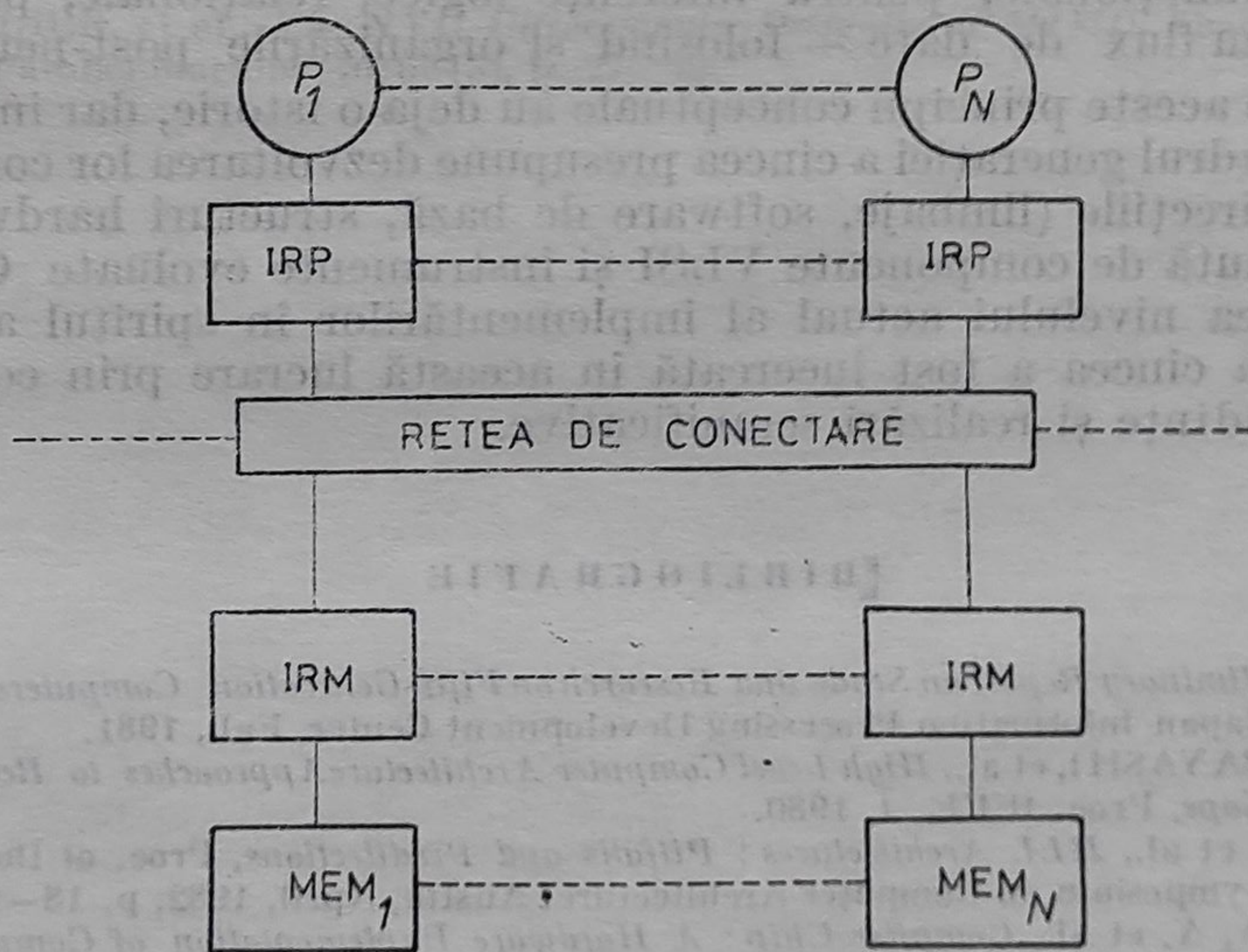


Fig. 5. — Sistemul NYU.

De fapt se obține interconectarea a  $N$  procesoare și a  $N$  module de memorie, printr-o rețea de comutatoare și interfețe rețea-procesoare (IRP), respectiv interfețe rețea-modul de memorie (IRM). Din punctul de vedere al costului și al performanței, rețeaua de comutatoare este dominantă. Pînă în anul 1990 se presupune că vor fi fezabile comutatoarele pe unul sau două cipuri, de  $4 \times 4$  linii bidirecționale, iar rețeaua de interconectare va fi dispusă pe 6 etaje. În cadrul simulărilor, fiecare procesor a fost considerat echivalentul unității centrale CDC 6600.

Pentru un sistem cu 4 096 de procesoare (exclusiv funcțiunile I/O) vor fi necesare 65 000 de cipuri, considerînd pentru anul 1990: 4 cipuri



pentru fiecare pereche procesor-IRP; 9 cipuri pentru fiecare pereche memorie-IRM (cipuri de 1 Mbit); două cipuri pentru fiecare comutator  $4 \times 4$ .

O alternativă la modelul NYU este oferită de arhitecturile modelelor cu flux de date (dataflow computers), dintre cele mai cunoscute cercetări fiind cele de la MIT, dar pînă în prezent nu a fost elaborat un proiect fezabil de implementare fizică.

## 7. Concluzii

Examinarea conceptelor arhitecturale privind generația a cincea de calculatoare prevăzute în proiectul japonez evidențiază accentul pus pe dezvoltarea unor sisteme de prelucrare a cunoștințelor cu echipamente dedicate, de mare productivitate. Discrepanța semantică existentă în prezent între nivelul hardware și utilizator va fi în mare măsură eliminată prin dezvoltarea unor interfețe inteligente. Software-ul de bază va fi organizat corespunzător celor trei componente esențiale ale sistemelor (gestiunea bazelor de cunoștințe, prelucrarea inferențială și interfața inteligentă), constituind legătura între interfața utilizator și echipamente. Structurile hardware care vor sta la baza configurațiilor vor fi de tipul mașinilor funcționale, pentru inferențe logice, relaționale, pentru date abstracte, cu flux de date — folosind și organizările post-neumanniene.

Toate aceste principii conceptuale au deja o istorie, dar implementarea lor în cadrul generației a cincea presupune dezvoltarea lor considerabilă pe toate direcțiile (limbaje, software de bază, structuri hardware), care va fi susținută de componente VLSI și instrumente evaluate CAD/CAM. Surprinderea nivelului actual al implementărilor în spiritul arhitecturii generației a cincea a fost încercată în această lucrare prin considerarea citorva tendințe și realizări semnificative.

## BIBLIOGRAFIE

1. \* \* \* *Preliminary Report on Study and Research on Fifth-Generation Computers*, 1979–1980, Japan Information Processing Development Center, Fall, 1981.
2. N. KAMIBAYASHI, et al., *High Level Computer Architecture Approaches to Reduce Semantic Gaps*, Proc. IEEE, 1, 1980.
3. K. KAVI, et al., *HLL Architectures: Pitfalls and Predilections*, Proc. of the 9th Annual Symposium on Computer Architecture, Austin, April, 1982, p. 18–23.
4. FUSAOKA, A. et al., *Compiler Chip: A Hardware Implementation of Compiler*, Proc. of the Symposium on Architectural Support for Programming Languages and Operating Systems, Palo Alto, March, 1982, p. 92–95.
5. J. HOLLOWAY, et al., *The SCHEME-79 Chip*, MIT AI Memo No. 559, January, 1980.
6. J. P. SANSONNET, et al., *Direct Execution of LISP on a LISP-Directed Architecture*, idem [4], p. 132–139.
7. Ed. FEIGENBAUM, et al., *The Fifth Generation*, Addison-Wesley, 1983.
8. P. TRELEAVEN, et al., *A Recursive Computer Architecture for VLSI*, idem [3], p. 229–238.
9. W. F. CLOCKSIN, et al., *Programming in Prolog*, Springer-Verlag, 1981.
10. \* \* \* *Development tool for Japan's 5th generation computer project*, Electronics, ian., 1984.
11. G. SANDBERG, *A primer on relational data base concepts*, IBM Systems Journal, 20, 1, 23–40 (1981).
12. M. W. BLASGEN, *System R: An architectural overview*, IBM Systems Journal, 20, 1, 41–62 (1981).



13. M. M. ZLOOF, *Office-by-Example: A business language that unifies data and word processing and electronic mail*, IBM Systems Journal, **21**, 3, 272—304 (1982).
14. S. A. SCHUSTER, et al., *RAP. 2 — An Associative Processor for Data-bases and Its Applications*, IEEE Trans. on Comp., **c-28**, 6, 446—458, (1979).
15. S.Y.W. SU, et al., *The Architectural Features and Implementation Techniques of the Multicell CASSM*, idem [14], p. 430—445.
16. D. J. DE WITT, *DIRECT — A Multiprocessor Organization for Supporting Relational Database Management Systems*, idem [14], p. 395—405.
17. Y. DOHI, *Hardware Sorter and Its Application to Data Base Machine*, idem [3], p. 218 — 225.
18. N. S. CHANG, et al., *Picture Query Languages for Data Base Systems*, Computer, nov. 23—33 (1981).
19. L. L. KINNEY, et al., *An architecture for a VHSIC Computer*, The 8th Annual Symposium on Computer Architecture, 1981.
20. S. E. MADNICK, *Recent Research Results on the INFOPLEX Intelligent Memory System (IMS) Project*, Proceedings of the International Congress on Applied Systems Research and Cybernetics, December, 1980.
21. MIHAI MÂRȘANU, *Modeling of rotational storage devices*, Invited talk given at MIT, Cambridge, U.S.A., May, 20, 1982.
22. MIHAI MÂRȘANU, *Modelarea și simularea performanțelor memoriilor externe rotaționale de mare densitate de tip disc magnetic*, Conferința Națională de Electronică, Telecomunicații, Automatică și Calculatoare, 15—17 noiembrie 1984, Institutul politehnic din București.
23. MIHAI MÂRȘANU, *Dispozitive funcționale cu structuri actuale și neconvenționale pentru optimizarea timpului de acces la date pe suporturi magnetice de tip disc*, Comunicare științifică propusă pentru prezentare la Conferința Națională de Cibernetică, București, 3—5 octombrie 1985.
24. A. GOTTLIEB, et al., *The NYU Ultracomputer-Designing a MIMD, Shared — Memory Parallel Machine*, idem [3], p. 27—42.



# GENERAȚIA A CINCEA DE CALCULATOARE ELECTRONICE ȘI INDUSTRIA DE TEHNICĂ DE CALCUL ROMÂNEASCĂ

VICTOR MEGHEȘAN\*)

NICOLAE COSTAKE\*\*)

ABOUT THE 5th GENERATION COMPUTERS AND THE ROMANIAN COMPUTER INDUSTRY. The project of the 5th generation computers outlined priority objectives for the development of the computer technique at the 1990...1995 levels. After a short strategy and resource evaluation and taking into account that the priority development objectives of every field depends also on the nature of the socio-economic system, viz., the national characteristics, general technical objectives are proposed for the development of the Romanian computer industry and the content of a programme is sketched in consideration of the objectives of the 5th generation computers.

În ultimul deceniu dezvoltarea tehnicii de calcul și utilizarea acesteia în Japonia s-a aflat în atenția specialiștilor din toate celelalte țări. Datorită caracterului său de sfidare tehnică, proiectul japonez al generației a cincea de calculatoare electronice [4] a trezit un viu interes (de exemplu [5]), desigur, și prin încercarea de a crea un nou grad de libertate tehnologică în sensul lui Ishigai [6]. Prezenta lucrare încearcă să pună în discuția specialiștilor câteva aspecte și propuneri care ar putea fi aplicate în vederea dezvoltării industriei de tehnică de calcul românească.

## 1. O scurtă caracterizare a proiectului generației a cincea de calculatoare electronice (Japonia)

Proiectul generației a cincea de calculatoare electronice a fost inițiat în anul 1979 de către Ministerul Comerțului Internațional și Industriei, în anul 1982 creindu-se institutul orientat pe acest proiect, avînd sarcina de a elabora pînă în anul 1990 prototipul, cu o finanțare de circa 850 mil \$ (din care 450 mil. \$ de la bugetul statului) și asigurînd colaborarea a mai multe mari firme japoneze interesate (printre care NEC, Fujitsu, Hitachi, Mitsubishi, Sharp, Toshiba). Obiectivul urmărit este cel de a cuceri locul conducător în tehnica de calcul pe plan mondial, avîndu-se în vedere atît creșterea eficienței economiei japoneze cît și o substanțială creștere a potențialului de export japonez în domeniul tehnicii de calcul.

*Din punctul de vedere al utilizatorului* specificul calculatorului de generația a cincea poate fi considerată inteligența artificială în sensul sistemului expert\*\*) și interfața bazată pe utilizarea limbajului natural, a

\*) Institutul de cercetare științifică și inginerie tehnologică pentru tehnică de calcul și informatică.

\*\*) În măsura în care accentul cade pe sisteme expert și nu pe sisteme creatoare (care generează automat concepte și proceduri noi) este poate mai propriu termenul de înțelepciune artificială.



simbolurilor și imaginilor, cunoștințele și procedurile experților fiind memorate în baza de cunoștințe (cu ajutorul specialiștilor în ingineria cunoașterii) și devenind astfel direct accesibile celui interesat (care este dispensat de asimilarea cunoștințelor de informatică sau de recurgerea la intermediul analistului/programatorului cu inerentele întârzieri și neînțelegeri). S-a considerat că această funcționalitate va permite, în principal :

- creșterea productivității muncii în ramurile primare (de ex : agricultură) și în ramurile indirect productive (de exemplu administrație), completând sistemele de conducere a proceselor tehnologice și sistemele de automatizare a fabricației și montajului din industrie ;

- consolidarea poziției în cadrul concurenței internaționale (de exemplu prin : creșterea productivității activității intelectuale, accelerarea cercetării științifice și dezvoltării tehnologice prin asistarea de inteligență artificială, traducerea automată a textelor și conversației) ;

- realizarea de economii de resurse (de exemplu prin : creșterea fiabilității și duratei de serviciu a echipamentelor ca urmare a cercetării, proiectării și fabricației asistată de calculator, minimizarea transportului zilnic al persoanelor prin distribuirea locurilor de muncă, inclusiv promovarea muncii din industria de informație la domiciliu) ;

- creșterea randamentului de utilizare a energiei și celorlalte resurse ;

- combaterea efectului de îmbătrânire a populației (de exemplu prin : informatizarea medicinei inclusiv pentru prelungirea vieții active).

*Din punct de vedere tehnic*, specificul calculatorului de generația a cincea poate fi considerat :

- *ca echipament* : utilizarea de circuite integrate pe scară extrem de largă (de ordinul a 1 mil. elemente/plachetă) ; realizarea de viteze de prelucrare extrem de ridicate, de ordinul a  $100 \div 1\,000$  mld operații/s ( $0,1 \div 1$  mld. inferențe logice/s), inclusiv prin adoptarea unei structuri logice noi a calculatorului, capabilă de realizarea prelucrării paralele a informației ; utilizarea de memorii de ordinul a  $100 \div 1\,000$  GB, cu un timp de acces logic de ordinul secunde, organizate ca baze de date relaționale ; procesoare de algebră relațională, procesoare inferențiale și alte tipuri de elemente care să permită trecerea de la prelucrarea numerică la prelucrarea simbolică, în vederea realizării recunoașterii problemelor, detectarea cunoștințelor utile rezolvării și construirii pas cu pas a raționamentului care să conducă la o soluție acceptabilă, eventual optimă ; interfețe inteligente realizând interacțiunea pe bază de limbaj natural, simboluri, imagini.

- *ca produse program* : subsistemul de gestiune a bazei de cunoștințe, subsistemul de rezolvare a problemelor (interfața/inferențe), subsistemul interfețelor inteligente, posibilitatea programării în limbaj de nivel foarte înalt ;

- *din punctul de vedere al suportului de dezvoltare* : instrumente program extrem de dezvoltate, sisteme puternice de gestiune a bazelor de date, rețele locale de teleprelucrare, rețele naționale de teleprelucrare.

*Din punct de vedere economic și organizatoric*, proiectul calculatorului de generația a cincea poate fi caracterizat prin următoarele :

- proiect prioritar pe termen lung inițiat de conducerea statului, care contribuie la finanțarea sa cu peste 50 %, caracteristicile de durată și resurse financiare (circa 8,5 mld. \$ an) fiind cele ale unui proiect de cercetare științifică și dezvoltare tehnologică mare, de importanță națională



(pentru comparație se citează cele circa 0,7 mld. \$ an aferente proiectului brațului canadian al navetei spațiale Challenger [7]):

- organizarea bazată pe crearea unei unități specializate relativ mici (circa 40 cercetători sub 35 ani recrutați din cadrul firmelor direct interesate în proiect), avînd pe de o parte suportul unor consilii științifice și consilieri pentru orientarea lucrărilor, pe de altă parte suportul colaborării cu principalele firme interesante pe bază contractuală, în concordanță cu propriile planuri de cercetare și dezvoltare tehnologică ale firmelor;

- măsurile deosebite luate pentru asigurarea transferului rapid al rezultatelor la firmele finanțatoare: stagiul în institutul de cercetări limitat la  $3 \div 4$  ani; raportarea periodică de către membrii unității specializate de cercetare — în principiu săptămînală — a programului realizat;

- în vederea realizării potențialului de export, elaborarea se bazează pe efortul național (propriu).

## **2. O scurtă încercare de evaluare a proiectului generației a cincea de calculatoare electronice în raport cu stadiul și cerințele dezvoltării economico-sociale a țării noastre**

Desigur că o bună parte din obiectivele și caracteristicile tehnice ale proiectului calculatorului de generația a cincea sînt interesante din punct de vedere științific, tehnic și economic și pentru țara noastră.

Se poate pune întrebarea dacă este tehnic și economic oportună abordarea unui proiect identic în țara noastră. Autorii prezentei lucrări împărtășesc părerea că răspunsul la această întrebare este negativ, cel puțin datorită următoarelor considerente:

a) Valoarea resurselor financiare alocate prin proiectul japonez cercetării și dezvoltării, reprezintă circa  $2/3$  din bugetul anual de cercetare și dezvoltare al firmei IBM.

b) Caracteristicile tehnice ale vitezei de prelucrare și memoriei preconizate pentru calculatorul de generația a cincea se situează cu cîteva ordine de mărime peste realizările medii actuale.

Rezultă că o eventuală abordare imediată a proiectului calculatorului generației a cincea, așa cum este definit de Japonia, nu poate constitui un obiectiv realist. Pe de altă parte, cercetarea științifică și dezvoltarea tehnologică de tehnică de calcul românească nu poate să facă abstracție de apariția în anii '90 a acestei noi generații de calculatoare. Rezultă că se impune o selecție de obiective, care să asigure deschiderea spre soluții care apar astăzi ca foarte avansate. Pentru aceasta, trebuie pornit în primul rînd de la sarcinile puse în fața tehnicii de calcul românești.



Din cuvântările președintelui Nicolae Ceaușescu, secretarul general al P.C.R., și din celelalte documente de partid, rezultă următoarele sarcini esențiale care revin domeniului producției și utilizării tehnicii de calcul :

- realizarea și dezvoltarea industriei de tehnică de calcul pe bază de cercetare proprie sau de cooperare în fabricație, în măsură să asimileze în fabricație produse de tehnică de calcul avansate (inclusiv sisteme de minicalculatoare, sisteme de microcalculatoare și alte produse bazate pe utilizarea microprocesoarelor) ;

- realizarea unui sistem informatic național, bazat pe produsele de tehnică de calcul din producția internă și cuprinzând între altele și rețeaua națională de teletransmitere a datelor și de calculatoare electronice, ca suport al perfecționării planificării și conducerii (în legătură cu aceste obiective a fost prevăzută și realizarea unui sistem informațional economico-social, modern, integrat) ;

- necesitatea creșterii substanțiale a contribuției tehnicii de calcul la mărirea productivității muncii și reducerea valorii cheltuielilor materiale ;

- creșterea ponderii produselor la nivel mondial sau depășind nivelul mediu mondial, ca bază a unei substanțiale mărimi a exportului de tehnică de calcul.

Este evident că tehnica de calcul poate contribui la rezolvarea problemelor puse de construcția economico-socială din țara noastră la un nivel tehnic ridicat și într-o perspectivă generală (și nu numai locală). În acest sens, de altfel, au fost preconizate direcțiile de dezvoltare pe termen mediu [8]. În pregătirea acestora au fost cercetate sistematic opiniile utilizatorilor precum și căile principale prin care tehnica de calcul poate contribui la creșterea eficienței activității economico-sociale, în special a creșterii productivității muncii sociale. Opiniile utilizatorilor de tehnică de calcul au fost evaluate între altele prin două anchete organizate de I.T.C.I. cu sprijinul conducerii ministerului tutelar.

Au rezultat următoarele concluzii principale, care exprimă problemele și tendințele actuale la nivelul întreprinderilor și a altor unități economico-sociale :

a) este necesară asigurarea echipamentelor și produselor program cu caracter de generalitate, capabile să constituie suportul unor sisteme informatice de mare eficiență ;

b) este necesar ca introducerea sistemelor de tehnică de calcul să se bazeze pe o concepție tehnică generală cunoscută și acceptată la nivel național ;

c) este necesară o îmbunătățire a informării tehnice cu privire la noile realizări ale producției de tehnică de calcul și la noile aplicații, a pregătirii informatice a utilizatorilor (în special a cadrelor de conducere).

Se constată că aceste concluzii au în vedere apropierea sistemelor de tehnică de calcul de utilizator. O analiză a efectelor potențiale ale utilizării tehnicii de calcul a condus la concluzia că acestea pot fi considerabile. Din punct de vedere calitativ, principalele aspecte sînt cele din tabelul 1.



Tabelul 1

O evaluare globală a surselor de efect economic ale introducerii tehnicii de calcul

Nr. crt.	Sursa de creștere a venitului național	Factori limitatori posibili	Nivel	Acțiuni posibile ale tehnicii de calcul	Condiții principale pentru obținerea eficienței
0	1	2	3	4	5
1.	Reducerea cheltuielilor materiale (pondera cheltuielilor materiale în produsul social este de ordinul a 55—60%)	a. Informarea tehnică și tehnologică insuficientă pentru elaborarea, proiectarea, utilizarea produselor, precum și pentru tipizare și standardizare	Toate	Rețea, bănci de date, produse/tehnologii	Nomenclator unitar produse/servicii/echipamente Nomenclator unitar tehnologii Sistem informațional național unitar al standardelor, normelor și brevetelor interne și internaționale Rețea națională de teleprelucrare
		b. Proiectarea optimă a produselor și tehnologiilor presupune un volum foarte ridicat de calcule, practic nerealizabile cu mijloace convenționale	Unități cercetare-proiectare	Proiectarea interactivă a produselor și tehnologiilor cu suportul calculatorului	Produse program și echipamente de capacitate suficientă, permițând lucrul interactiv grafic
		c. necunoașterea consumului cumulat de resurse materiale, energie și import la nivelul produselor face ca deciziile referitoare la reducerea cheltuielilor materiale să nu fie, în general, optime	Intreprinderi, centrale, ministere	Calculul periodic al consumului cumulat de resurse primare la nivelul principalelor grupe de produse	— Modele macroeconomice — Sistem informațional statistic național integrat
		d. valorile componentelor costurilor unitare de producție nu sînt cunoscute operativ, scăzînd eficacitatea acțiunilor proiectanților	Intreprinderi, unități de proiectare	Elaborarea automată a analizei consumurilor specifice și ale componentelor costurilor de producție pe produse	— Baza de date produse
		e. necunoașterea precisă a fluxurilor materiale nu permite identificarea unor surse importante de pierderi	Intreprinderi și unități similare	Cîntărirea automată a materialelor + calculul automat al balanțelor materiale etc.	— Sisteme de cîntărire automate și de prelucrare a datelor pe flux



0	1	2	3	4	5
		f. fluctuațiile de calitate generează rebuturi, acțiunile corectoare nefiind posibile fără suportul prelucrării statistico-matematice a datelor de calitate a produselor	Intreprinderi și unități similare	Promovarea ingineriei calității asistate de calculator	— Organizarea corespunzătoare a evidenței CTC
2.	Reducerea stocurilor materiale (utilizarea informaticii are, în general, ca efect o reducere cu 2÷5 % a valorii stocurilor medii)	a. utilizarea în produse a unei game largi de tipodimensiuni de materiale consumate în cantități reduse, crește probabilitatea stocurilor cu mișcare lentă b. dimensionarea stocurilor pe criterii de cost minim sau probabilitate de ruptură de stoc minimă necesită calcule statistico-matematice permanente c. în lipsa unui serviciu de informare de excepție rapid și sigur, pentru a face față urgențelor, stocurile de siguranță se dimensionează, cu necesitate, la un nivel relativ ridicat	Baze și unități de aprovizionare, întreprinderi  Baze și unități de aprovizionare întreprinderi, mari unități comerciale  Toate unitățile	Identificarea consumurilor reduse ca suport al recomandărilor de standardizare și /sau tipizare (informarea proiectanților)  Conducerea optimă a stocurilor  Serviciu de teleprelucrare a informației de excepție de tip „necesar urgent/disponibil imediat”	Sistem teleprelucrare  Sisteme de conducere cu calculator sau minicalculator și terminale industriale Legiferarea inventarului permanent  Rețea națională de teleprelucrare Nomenclator unitar de produse/servicii/echipamente Nomenclator unitar al unităților economico-sociale
3.	Reducerea valorii fondurilor fixe neintrate în funcțiune (în condițiile creșterii economice și a întârzierii de 1—2 ani în realizarea investițiilor, valoarea mijloacelor fixe imobilizate poate atinge fracții semnificative	Numărul mare de activități intercondiționate face imposibilă cuprinderea lor cu mijloace convenționale și implicit alocarea națională a resurselor pentru investiții	Unități de proiectare Unități de construcții Unități coordonatoare	Utilizarea metodelor matematice de planificare și urmărire, bazate pe teoria grafurilor și programare matematică	Baza de date a obiectivelor de investiții



(continuare tabelul 1)

0	1	2	3	4	5
	din valoarea venitului național)				
4	Creșterea preciziei planificării (precizia este de ordinul a $0,5 \div 3\%$ )	Considerarea separată a activităților, mai ales în condițiile unei producții diversificate poate conduce la necorelări considerarea lor integrată nefiind posibilă cu metode convenționale	Unități tip întreprinderi	Analiza și conducerea integrată a activităților tehnice și economice	Echipamente și produse program pentru conducerea integrată a producției
		Dimensiunile problemei planificării au devenit foarte mari, utilizarea metodelor convenționale implicând aproximații și simplificări	Unități tip centrală industrială	Utilizarea modelării macroeconomice	Bază de serii de date macroeconomice
		În condițiile actuale ale perturbațiilor care se manifestă în economia mondială, metodele convenționale de planificare nu pot asigura viteza de răspuns și generarea variantelor necesare luării deciziilor de compensare a efectului perturbațiilor	Organe centrale și teritoriale		Bază de meta-date
			Unități tip centrală industrială	Utilizarea metodelor matematice de analiză, prognoză și optimizare	Perfecționarea metodologiilor de luare a deciziilor
			Organe centrale și teritoriale		Modelare matematică
			Unități de comerț exterior		
5	Creșterea eficienței utilizării resurselor materiale (utilizarea conducerii automate proceselor tehnologice permite o creștere a eficienței fondurilor fixe cu cel puțin $1\%$ )	Conducerea proceselor tehnologice cu metode convenționale nu permite realizarea unei producții și calități constante	Unități tip întreprinderi	Conducerea automată a proceselor tehnologice	Definirea și măsura calității. Elemente de automatizare cu utilizarea de sisteme cu fiabilitate ridicată



(continuare tabelul 1)

0	1	2	3	4	5
		Deservirea/conducerea utilajelor de către operatorul uman este afectată de factori subiectivi (neatenție, oboseală, boală) și este supusă limitărilor caracteristice performanțelor operatorului uman	Unități tip întreprindere	Conducere automată a utilajelor Robotizare	
		Procesul de învățare a noilor deprinderi și/sau tehnologii este relativ lent și însoțit de durate importante de funcționare cu eficiență redusă	Toate unitățile	Creșterea calității instruirii și a personalizării ei prin instruire programată precum și prin simulatoare	Sisteme cu acces multiplu  Produse program specifice
			Populația	Încurajarea utilizării tehnicii de calcul prin asimilarea ușoară și plăcută	Calculatoare individuale și de casă
		Realizarea produselor tehnologice în modul cel mai eficient tehnic și economic implică cunoașterea posibilităților existente, evitând investiții în utilaje relativ necritice	Unități tip întreprindere	Serviciu de teleinformare capacități de producție/operații tehnologice realizabile imediat, necesare urgent	Rețea de teleprelucrare
		Asigurarea structurii optime pe specialități a absolvenților învățământului tehnic în condițiile revoluției tehnice și științifice contemporane necesită informa-	Toate unitățile și organele centrale cu atribute de planificare a muncii și pregătirii profesionale	Modelarea procesului de instruire și încadrarea sa în modelul dezvoltării sistemului economic	Sistem informațional integrat



(continuare tabelul 1)

0	1	2	3	4	5
6	Creșterea valorii exportului (soldul balanței comerțului exterior cunoaște fluctuații, mai ales în condițiile perturbațiilor actuale ale economiei actuale, creșteri ale valorii exportului cu 5% apărând ca obiective realizate)	<p>ții multiple, inclusiv de ansamblu precum și calcule greu de efectuat cu metode convenționale</p> <p>Necunoașterea conținutului cumulat de import în produse/servicii nu permite calculul exact al eficienței exportului (acest calcul nu este posibil decât prin modelarea la nivel macroeconomic)</p> <p>În condițiile revoluției tehnice și științifice contemporane, prețul produselor crește cu gradul de electronizare respectiv de implicare a tehnicii de calcul</p> <p>Probabilitatea de încheiere de contracte avantajoase crește cu viteza de răspuns, inclusiv acceptarea de clauze de compensare prin diferite tipuri de mărfuri/servicii</p> <p>Realizarea și păstrarea poziției în comerț depinde și de cunoașterea concurenței și partenerilor</p>	<p>Unități cu activitate de comerț exterior</p> <p>Întreprinderi</p> <p>Unități cu activitate de comerț exterior</p> <p>Unități cu activitate de comerț exterior</p>	<p>Calculul conținutului cumulat de import pe unitatea de produs/serviciu</p> <p>Electronizarea utilajelor. Creșterea subramurii industriale de produse-program, inclusiv pentru export</p> <p>Sistem informatic integrat al activității de comerț exterior, inclusiv automatizarea activității de birou aferente</p> <p>Bază de date concurenți/parteneri</p>	<p>Sistem informațional statistic integrat actualizând o bază de date a produselor/serviciilor</p> <p>Sistem informațional integrat</p> <p>Organizația suport tip documentare și sistem integrat de informare</p>



Evaluarea cantitativă prin calcule orientative [11] a condus la următoarele concluzii :

a) Efectul economic realizabil depinde de mărimea, natura și condițiile dotării cu tehnică de calcul:

Opțiune	Valoarea totală a dotării/valoarea produsului social, circa	Ordinul de mărime al efectului economic (creșterea peste plan a venitului național anual/produs social circa)
Dotare minimă	1,5%	0,3%
Dotare completă, sistem integrat	5%	3%

b) Pentru a obține eficiența economică maximă a utilizării tehnicii de calcul, sînt necesare acțiuni corelate de introducere a tehnicii de calcul și de perfecționare a sistemului informațional la toate nivelurile și în toate domeniile (chiar dacă principalul efect direct este de așteptat să provină de la nivelul întreprinderilor industriale : circa 60%).

Din cele de mai sus, rezultă că pentru țara noastră cu toate că, în general, obiectivele avute în vedere la lansarea generației a cincea sînt valabile (de exemplu cel al apropierei nemijlocite ale utilizărilor, cel al întăririi poziției pe piața internațională și cel al economisirii resurselor, prezintă o evidentă actualitate), în etapa actuală obiectivul esențial al dezvoltării și utilizării tehnicii de calcul este realizarea eficienței în sensul creșterii mai rapide a venitului național și a productivității muncii sociale, pornind de la prevederile programului elaborat din inițiativa și sub directa îndrumare a tovarășului Nicolae Ceaușescu. Desigur că această dezvoltare trebuie să se realizeze compatibil cu elaborarea generației cinci de calculatoare electronice. Are de aceea sens definirea „proiectului de tip generația cinci” ca ansamblul obiectivelor de cercetare-dezvoltare în perspectivă a tehnicii de calcul, elaborate pe baza analizei necesităților și posibilităților concrete și ținînd seama de tendințele pe plan mondial.

### 3. O încercare de definire „a proiectului tip generația cinci”

Se propune ca obiectivul general al „proiectului tip generația cinci” să îl constituie creșterea suplimentară a venitului național și implicit a productivității muncii sociale pe baza utilizării tehnicii de calcul într-un mod compatibil cu utilizarea viitoarelor calculatoare de generația cinci.

Se propune ca din acest obiectiv general să fie derivate următoarele :

— Crearea premiselor generale necesare :

• realizarea compatibilității cu produsele existente pe plan mondial (în primul rînd cu seriile unitare de calculatoare/minicalculatoare/micro-



calculatoare din țările socialiste) deci aplicarea consecventă a standardelor ISO și CAER precum și a normativelor CIRC.

Se au în vedere atât lărgirea posibilităților de colaborare și cooperare internațională (inclusiv de creștere a exportului) precum și economia de resurse umane limitate prin evitarea elaborărilor paralele sau lipsite de potențial de export.

- realizarea sistemului informațional economico-social perfecționat pe baza prevederilor programului aprobat de conducerea superioară de partid, cu suportul unui sistem informatic de informare și analiză macroeconomică unitar;

- realizarea sistemelor standard modulare pentru conducerea proceselor tehnologice și a producției.

- Realizarea unor proiecte de natură a realiza un suport pentru utilizarea calculatoarelor de generația cinci și totodată pentru producția destinată consumului intern și exportului.

- Realizarea unei baze de cercetare pentru menținerea contactului cu proiectele avansate pe plan mondial în domeniul generației cinci.

Pe baza celor expuse, se poate propune ca „proiectul de tip generația cinci” al R. S. România să fie definit ca ansamblul a două categorii de subproiecte :

a) Subproiecte de natura dezvoltării evolutive a realizărilor existente, care să contribuie la crearea cadrului în care implementarea calculatoarelor de generația a cincea devine posibilă :

- subproiectul „sistemul unitar al produselor de tehnică de calcul”, al cărui rezultat așteptat este familia unitară a produselor de tehnică de calcul (sisteme, echipamente, programe) produse în țară, care să asigure compatibilitatea cu principalele produse elaborate în țările avansate industriale și capabilă să satisfacă cerințele de realizare a celor mai uzuale sisteme de conducere a proceselor tehnologice, ale automatizării și roboticii industriale, a sistemelor de conducere a unităților, a subsistemelor sistemului informatic național cinci ;

- subproiectul „informatica macroeconomică”, al cărui rezultat așteptat este ansamblul bazei unitare de metadate de interes național, al băncilor de date de interes național, al sistemului informatic unitar de informare și urmărire a realizării planului, al sistemului informatic al modelelor macroeconomice de interes național, departamental și teritorial ;

- subproiectul „instruire asistată de calculator” al cărui rezultat așteptat este un sistem interactiv de instruire programată cu acces multiplu la o bază de date de capacitate medie (minim 1 000 MB) ;

- subproiectul „elemente cu grad de integrare foarte ridicat”, al cărui rezultat așteptat este recomandarea soluțiilor tehnice și tehnologice pentru industria națională în acest domeniu ;

- subproiectul „memorii de mare capacitate” al cărui rezultat așteptat este elaborarea de prototipuri de unități de memorie externă cu acces direct de tip convențional și asociativ și a recomandărilor privind introducerea în fabricație.

b) Subproiecte de natura unor subproiecte ale proiectului japonez, care prezintă actualitate și au șanse de realizare în condițiile țării noastre (tabelul 2) :

- subproiectul „arhitectura calculatoarelor electronice”, al cărui rezultat așteptat este recomandarea unor prototipuri cu structuri fiabile capabile să realizeze viteze de prelucrare a informației foarte ridicate, utili-



Tabelul 2

Scurtă evaluare a subproiectelor definite în proiectul japonez al calculatoarelor de generația a cincea

Nr. crt.	Subproiectul definit în cadrul proiectului japonez (etapa inițială)	Propunere de includere în proiectul „tip generația cincea”	Comentarii
1	2	3	4
1.	Arhitectura distribuită funcțional	Cercetarea și dezvoltarea arhitecturii rețelelor : standarde (incl. protocoale), produse pentru interconectare, produse-program pentru prelucrare distribuită, produse pentru asigurarea confidențialității/secretizării, rețele locale, comunicații prin fibre optice. Acestea reprezintă premisa necesară realizării sistemelor informatice de înaltă eficiență, independent de proiectul generației a cincea	Proiectul japonez ia în considerare și problema telecomunicațiilor prin satelit, a elaborării de sisteme de operare pentru rețele și alte aspecte
2.	Calculator PROLOG	Cercetarea arhitecturilor noi ale calculatoarelor electronice (soluții necesare realizării de calculatoare cu performanțe ridicate în condițiile utilizării de elemente de viteză relativ redusă)	Eventualele prototipuri noi ce se vor realiza (de ex. calculator LISP) vor putea fi asimilate prin microproducție
3.	Calculator bază de date relațională		
4.	Calculator tip flux de date		
5.	Calculator LISP		
6.	Calculator cu structură tip post von Neumann		
7.	Circuite cu grad de integrare foarte ridicat		Limitările tehnice și tehnologice existente nu permit a se întrevăde posibilitatea realizării într-un timp scurt (proiectarea circuitelor larg integrate suportul calculatorului, inclusiv pentru cerințe specifice și urmărirea realizărilor avansate din alte țări sînt în mod evident necesare)
8.	Produse-program de gestiune pentru baze de cunoștințe	subproiect „intelligență artificială” în sensul realizării unui sistem expert de capacitate redusă găzduit de calculatoare existente (evaluarea problemelor ingineriei cunostințelor)	Interesul practic principal îl constituie sistemele expert pentru proiectare asistată de calculator (de ex. circuite electronice, echipamente)



(continuare Tabelul 2)

1	2	3	4
9.	Produse-program pentru rezolvarea problemelor (și inferențiale)		
10.	Sisteme pentru comunicare om-calculator	elaborarea soluțiilor pentru intrări/ieșiri neconvenționale (produsele pentru intrare/ieșire grafică/imagini respectiv locală sînt necesare pentru utilizarea eficientă a tehnicii de calcul în proiectarea asistată de calculator, sisteme informatice pentru conducere, conducerea automată a proceselor tehnologice, robotică)	
11.	Produse-program pentru interfețe inteligente		
12.	Produse-program pentru traducere automată		
13.	Produse-program întrebare/răspuns	se consideră parte a subproiectului „inteligentă artificială” (evaluarea problemelor de elaborare a produselor-program de utilizare a unei baze de cunoștințe	
14.	Produse-program de recunoaștere a vorbirii	se consideră parte a subproiectului „intrări/ieșiri neconvenționale” și a subproiectului „limbaje de programare evolute”	
15.	Produse-program pentru recunoașterea figurilor și imaginilor		
16.	Produse-program pentru rezolvarea automată a problemelor de tip matematic		
17.	Sistem expert cu caracter de suport pentru dezvoltare		În funcție de rezultatele proiectului „inteligentă artificială” este posibil să se realizeze cel puțin parțial, acest obiectiv
18.	Sistem de programe pentru dezvoltarea produselor-program	Elaborarea produselor-program complexe asistată de calculator (soluție eficientă pentru creșterea productivității muncii de proiectare a produselor-program)	Crearea unui astfel de instrument este de natură să permită abordarea elaborării de sisteme de operare proprii (compatibile UNIX)



(continuare Tabelul 2)

1	2	3	4
19	Baze de date pentru dezvoltare		
20	Precizarea interfețelor dintre toate componentele proiectului		
21	Elaborarea instrumentelor de măsurare a performanțelor		Necesitatea unor asemenea produse este generală, nu numai pentru calculatoarele de generația cinci
22	Calculator personal pentru dezvoltare		
23	Rețea locală ca suport de dezvoltare	Considerată ca parte a proiectului „rețele de teleprelucrare”	
24	Rețea națională pentru suport de dezvoltare	Considerată ca parte a proiectului „rețele de teleprelucrare”	
25	Sistem pentru proiectarea circuitelor integrate pe scară foarte largă	Sisteme de proiectare asistate de calculator, inclusiv orientate pe proiectarea circuitelor larg și foarte larg integrate (creșterea productivității muncii de cercetare-proiectare)	
26	Sistem de gestiune pentru bază de cunoștințe	Elaborarea unui sistem de gestiune pentru baze de date relaționale (produs-program utilizabil pentru calculatoarele actuale și în același timp componentă a generației a cincea)	

zind elemente relativ lente (inclusiv arhitecturi multiprocesor) care să stea la baza proiectării viitoarelor minicalculatoare ce se vor introduce în fabricație;

- *subproiectul „rețele de teleprelucrare”* al cărui rezultat așteptat este intrarea în exploatare curentă a rețelei naționale de teletransmisie și teleprelucrare precum și sisteme standard de rețele locale;

- *subproiectul „intrări/ieșiri neconvenționale”*, al cărui rezultat așteptat sînt sisteme de intrare/ieșire vocale și grafice/imagini.

- *subproiectul „baze de date relaționale”*, al cărui rezultat așteptat este un sistem portabil de gestiune a bazelor de date convenționale/relaționale, destinat și băncilor de date ale sistemului informatic național, cu facilități de exploatare lot și interactivă;

- *subproiectul „sisteme de proiectare cu suportul calculatorului”*, al cărui rezultat așteptat este un număr de sisteme interactive orientate, implementate pe minicalculator de capacitate ridicată pentru domenii prioritare: proiectarea circuitelor electronice larg și foarte larg integrate, proiectarea plăcilor cu circuite imprimate, proiectarea echipamentelor electronice, proiectarea echipamentelor mecanice;



• subproiectul „elaborarea produselor program complexe cu suportul calculatorului”, al cărui rezultat aşteptat este un sistem orientat pe elaborarea produselor-program complexe (cu ajutorul cărui să fie elaborate într-o primă etapă sisteme de operare compatibile UNIX şi translatoare ADA sau alt limbaj de mare perspectivă);

• subproiectul „inteligenţă artificială”, al cărui rezultat aşteptat este realizarea, experimentarea şi evaluarea unui sistem expert de capacitate redusă (inclusiv a problemelor ingineriei cunoştinţelor);

• subproiectul „limbaje de programare evolute”, al cărui rezultat aşteptat este elaborarea unui translator interactiv capabil să accepte o intrare cât mai apropiată de limbajul natural.

#### 4. Observaţii finale

Se fac următoarele observaţii finale :

a) selecţia subproiectelor „de tip generaţia cinci” constituie o bază de discuţie supusă dezbaterii specialiştirilor ;

b) în prezenta comunicare nu se analizează problemele de natură economico-organizatorică, considerînd că acestea trebuie discutate într-o etapă ulterioară.

#### BIBLIOGRAFIE

- NICOLAE CEAUŞESCU, *Raportul Comitetului Central cu privire la activitatea Partidului Comunist Român în perioada dintre Congresul al XII-lea şi Congresul al XIII-lea şi activitatea de viitor a partidului în vederea îndeplinirii obiectivelor dezvoltării economico-sociale în cincinalul 1986 — 1990 şi, în perspectivă, pînă în anul 2000, a României*, Edit. politică, Bucureşti, 1984.
- \* \* \* *Directivile Congresului al XIII-lea al Partidului Comunist Român cu privire la dezvoltarea economico-socială a României în cincinalul 1986 — 1990 şi orientările de perspectivă pînă în anul 2000*, Edit. politică, Bucureşti, 1984.
1. V. BALTAC, A. DAVIDOVICIU, C. MASCHEK, S. DIACONESCU, M. SĂDEANU, M. MARTINOVICI, C. STĂNCESCU, *Sisteme interactive, limbaje conversaţionale*, Edit. politică, Bucureşti, 1984.
  2. M. DRĂGĂNESCU, *Realizări şi perspective în informatică. Informatica şi dezvoltarea economico-socială a României*, ICI, Bucureşti, 1983.
  3. V. BALTAC, D. ROMAN, C. ZEGHERU, D. COSTĂCHESCU, A. LISTIG, C. DUMITRESCU, C. STĂNESCU, V. TEPELEA, G. CORCODEL, M. JACOBESCU, *Calculatoarele electronice, grafică interactivă şi prelucrarea imaginilor*, Edit. tehnică, Bucureşti, 1985.
  4. \* \* \* *Preliminary report on study and research on 5th generation computers 1979—1980*, Japan's Info. Proc. Dev. Center., 1981.
  5. E. FEIGENBAUM, PAMELA Mc. CORDUCK, *The fifth generation*, Edit. Addison-Wesley Reading, Mass, 1983.
  6. S. ISHIGAI, *Legile interne de dezvoltare a tehnologiei* (un rezumat al cercetărilor din Japonia, în : *Ştiinţa tehnologie, dezvoltare socială şi umană*, Edit. politică, Bucureşti, 1984).
  7. W. ATKINSON, *Canada — magna cum laude*, Science Dimension (1982).
  8. V. MEGHIEŞAN, N. COSTAKE, M. MĂRŞANU, *Directivile de dezvoltare ale tehnicii de calcul româneşti la orizontul 1990* (vezi p. 75 din acest volum).
  9. V. MEGHIEŞAN, N. COSTAKE, M. MĂRŞANU, M. RIGANI, C. CAZAN, *Unele aspecte ale utilizării tehnicii de calcul în întreprinderile din cadrul MIMUEE*, Documentaţia, cod 7506/A, ITCI, Bucureşti, 1983.
  10. N. COSTAKE, M. MĂRŞANU, C. CĂLINESCU, C. VĂCEANU, *Ancheta efectuată la unităţile economico-sociale cu privire la problemele dezvoltării domeniului tehnicii de calcul*, Docum. cod PTK, ITC, Bucureşti, 1983.
  11. N. COSTAKE, M. MĂRŞANU, V. MĂNOIU, C. CĂLINESCU, C. VĂCEANU, *Program de introducere accelerată a automatizării şi electronicii în economia naţională — contribuţia tehnicii de calcul*, Documentaţia cod PTK ITC, Bucureşti, 1983.



## DIRECȚIILE DE DEZVOLTARE ALE TEHNICII DE CALCUL ROMÂNEȘTI LA ORIZONTUL 1990

VICTOR MEGHEȘAN\*), NICOLAE COSTAKE\*), MIHAI MÂRȘANU\*)

DIRECTIONS IN THE DEVELOPMENT OF COMPUTER INDUSTRY OF ROMANIA AT THE 1990 HORIZON. Some results of the studies and researches performed at the Institute of Computer Technique and Informatics (ITCI) aimed at outlining a development plan for the computer technique field in Romania at the 1990 horizon are presented.

Several suggestions are made to be further discussed by the specialists.

În vederea pregătirii proiectului de dezvoltare a tehnicii de calcul la orizontul 1990 au fost elaborate mai multe lucrări cu contribuția I.T.C.I. și a celorlalte unități de profil din cadrul ministerului.

S-a pornit de la orientările referitoare la domeniul tehnicii de calcul, cuprinse în cuvântările Secretarului general al P.C.R., tovarășul Nicolae Ceaușescu și în celelalte documente de partid. Au fost efectuate calcule de dimensionare orientative, avînd în vedere și analiza tendințelor pe plan mondial, precum și anchete la nivelul specialiștilor și unităților utilizatoare de tehnică de calcul.

### 1. Scurtă prezentare a principalelor probleme actuale

Principalele aspecte care caracterizează stadiul actual al producției și utilizării tehnicii de calcul în țara noastră pot fi considerate următoarele :

a) Succesele obținute (în special în asimilarea pe bază de concepție proprie a unor familii de minicalculatoare, de microcalculatoare, de terminale, de modem-uri, periferice etc.) și nivelul actual al cercetărilor și al producției ilustrează posibilitățile de accelerare a dezvoltării domeniului.

b) Analiza comparativă cu alte țări a nivelului cantitativ și calitativ reflectă o serie de deosebiri de structură, date în tabelul de mai jos :

---

\*) Institutul de cercetare științifică și inginerie tehnologică pentru tehnică de calcul și informatică.



Valori estimative la începutul acestui deceniu

Nr. crt.	Indicator (relativ prin raportarea la nivelul R.S.R.)	R.S.R.	Țări capitaliste dezvoltate industrial (S.U.A., Japonia, Franța)
1.	Tipul de prelucrare preponderent în economie	lot	interactiv
2.	Modul preponderent de culegere/furnizare a informației	cartela imprimat	Terminal conectat la rețea de teleprelucrare
3.	Raportul memorie*) externă pe disc (MB)/memorie internă (KB)	1	× 5
4.	Numărul calculatoarelor de capacitate mare, pe 1 milion locuitori*)	1	× 250
5.	Proporția numărului terminalelor conectate la linii de telecomunicații, pe 1000 locuitori*)	1	× 100

\*) După evaluarea autorilor

c) Cu toate succesele incontestabile realizate în utilizarea tehnicii de calcul în conducerea proceselor tehnologice și a întreprinderilor [4,6], este de așteptat o sporire considerabilă a eficienței utilizării tehnicii de calcul. Aceasta s-ar putea obține în mare parte, rezolvând următoarele probleme :

- principala capacitate de prelucrare electronică din economie este constituită încă de calculatoarele Felix C-256/512/1024, utilizate predominant în prelucrări de tip lot, ale căror caracteristici de performanță și fiabilitate sînt astăzi depășite\*\*);

- majoritatea configurațiilor de calculatoare și minicalculatoare sînt subdimensionate mai ales din punctul de vedere al memoriei externe și terminalelor, limitînd posibilitățile de exploatare eficientă :

- înzestrarea încă insuficientă a mini și microcalculatoarelor cu produse-program de aplicație cu caracter general, capabile să fie imediat utilizate în analiza și conducerea activităților întreprinderilor (baze de date, sisteme pentru conducerea producției etc.) ;

- lipsa terminalelor de tip industrial (de ex. pontaj/control acces, culegere date de producție), respectiv economico-financiar (de exemplu : tip terminal pentru operații financiare), limitează serios posibilitatea realizării legăturilor eficiente utilizator/calculator.

## 2. Strategia propusă pentru etapa 1985 — 1990

În lucrările citate este propusă o strategie bazată pe următoarele :

a) *Perfecționarea concepției tehnice unitare de utilizare a tehnicii de calcul* (în sensul sistemului informatic național), orientată pe :

(i) *structura funcțională* (fig. 1) : sistemul produselor de tehnică de calcul care realizează :

\* *la nivelul locurilor de muncă și echipamentelor tehnologice* : electro-nizarea/conducerea automată a echipamentelor, inclusiv conducerea nu-

\*\*) De remarcă că prin utilizarea sistemului de operare HELIOS se poate crește sim-țitor gradul de utilizare intensivă a configurațiilor FELIX.



merică a mașinilor unelte și robotizarea ; conducerea automată a proceselor tehnologice ; automatizarea controlului de calitate pe fluxul de producție ; automatizarea unor activități prioritare (proiectarea produselor și tehnologiilor, reducerea consumurilor materiale și energetice etc.) ; asigurarea informatică a locurilor de muncă (culegere/validare/transmitere de date și texte, secretariat automat, acces de la distanță la calculatoare).

La acest nivel principala dotare se referă la sisteme specializate („la cheie”), sisteme modulare, terminale și/sau microcalculatoare de utilizare individuală cu facilități de conectare la rețele locale.

\* *la nivelul unităților* : analiza și conducerea unor activități prioritare ; analiza și conducerea activităților secțiilor ; analiza și conducerea activităților subunităților și a unităților în ansamblu.

La acest nivel dotarea include sisteme de conducere bazate pe minicalculatoare și/sau calculatoare (eventual în structuri ierarhice) cu posibilitatea interfatăării cu rețele de teleprelucrare locale sau departamentale/naționale.

\* *la alte niveluri* : analiza și planificarea activităților ; pregătirea deciziilor de conducere ; în general memorarea/transmiterea/prelucrarea automată a informației prin utilizarea de rețele de calculatoare.

(ii) *structura de echipamente* : sistemul echipamentelor de tehnică de calcul și subansamblelor acestora (URC) sub forma unui sortiment restrâns dar suficient, de elemente tipizate care să permită realizarea subsistemelor/sistemelor prevăzute în structura funcțională (fig. 1). În figura 2 se sugerează structura sistemului echipamentelor și subansamblelor acestora de tehnică de calcul de destinație generală. Se are în vedere realizarea unor sisteme flexibile conform necesităților aplicațiilor concrete.

Schema din figura 2 are un caracter orientativ, urmînd a fi precizată conform unui proiect tehnic.

(iii) *structura produselor program* : sistemul produselor program de interes general, și al sistemelor de operare modulare. (Noțiunea de sistem de operare este înțeleasă în accepțiunea propusă de V. Baltac [2]). Structura este sugerată în figura 3, care are, de asemenea, un caracter orientativ.

(iv) *ansamblul interfețelor standard* : se prevăd trei tipuri de interfețe standard :

— *interfața standard infologică* (cu utilizatorul) : convențiile de definire a parametrilor de comandă a prelucrărilor lot ; convențiile de definire a interfeței prietenoase a programelor interactive ; standardizarea limbajelor de programare conform ISO, cu facilități de generare programe interactive multi-terminal FORTRAN/COBOL/BASIC ; standardizarea tastaturilor conform ISO/CAER/CITC ; definirea unor reguli generale de amplasare și marcare funcție butoane (compatibila CITC) ; documentația tehnică obligatorie definită ca sortiment și conținut de CAER/CITC ;

— *interfața standard datologică* (schimb date) : asigurarea schimbului de date prin fișiere bandă magnetică 1 600 bpi PE, disc flexibil, opțional casetă magnetică, avînd organizări standardizate, în plus fiecare sistem trebuie să admită opțional interfață de telecomunicație ; fiecare produs program trebuie realizat în versiune lot și interactivă (dacă ambele versiuni au sens tehnic) și elaborat/generat în versiuni executabile pe toate tipurile de calculatoare, respectiv mini/microcalculatoare aparținînd sistemului (la nivelul funcțiilor care au asigurare tehnică) ;



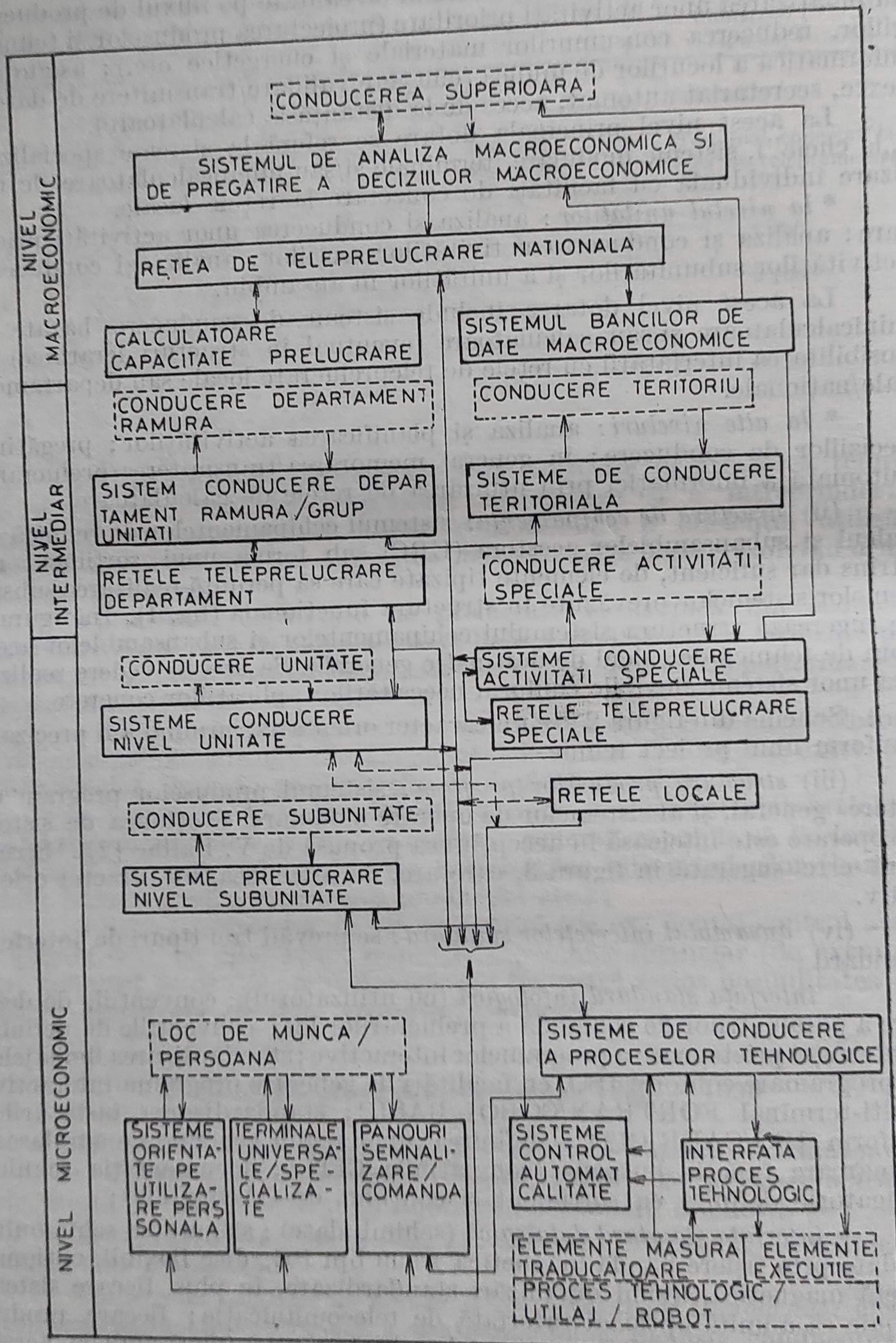


Fig. 1. — Structura funcțională a sistemului informatic general.











— *interfața standard tehnologică* (legături la nivelul echipamentelor) : standardizarea magistralelor, interfețelor și protocoalelor de telecomunicație/rețele conform ISO/CCITT/ CAER/CITO ; asigurarea prin cuploare respectiv drivere a posibilității conectării oricărui tip de unitate periferică la orice tip de unitate centrală (dacă această conectare are sens tehnic) ; standardizarea elementelor constructive conform standardelor și normelor ISO/CAER/CITO ; adoptarea standardelor mondiale pentru sistemele de operare ale minicalculatoarelor și microcalculatoarelor.

Definirea ansamblului interfețelor standard are în vedere alinierea la standardele/normele tehnice ISO/CAER/CITO, aliniere evident necesară din considerente de creștere a potențialului de export precum și a disponibilității.

b) *Dezvoltarea producției de tehnică de calcul prin asimilarea unor produse conform tendințelor verificate pe plan mondial [5]*

Astfel, s-au prevăzut :

- dezvoltarea producției de calculatoare electronice prin asigurarea compatibilității ES/IBM și orientarea pe teleprelucrare ;

- elaborarea și asimilarea în producție a minicalculatoarelor mono-placă de conducere a proceselor tehnologice, a minicalculatoarelor de mare productivitate de 16 biți, respectiv de 32 biți.

- producția unei game lărgite de unități periferice inclusiv unități cu discuri magnetice, unități cu bandă magnetică, terminale grafice alb-negru și color, terminale industriale (inclusiv portabile), terminale pentru operații economice și financiare etc. ;

- producția de unități specializate (minicalculator frontal, minicalculator versiune nod de telecomunicații, procesoare specializate) ;

- producția unei game de echipamente bazate pe microprocesoare/minicalculatoare pentru prelucrările de date cu caracter local (inclusiv rețele locale) ;

- elaborarea și asimilarea în producție a microcalculatoarelor individuale/familiale ;

- elaborarea de produse program cu performanțe ridicate (de exemplu : compilatorul LISP, baze de date pentru micro și minicalculatoare) ;

- dezvoltarea de sisteme specializate pentru domenii de larg interes (de exemplu : proiectarea interactivă automată [3]) ;

- abordarea cercetărilor în unele domenii de perspectivă ca de exemplu structuri avansate de calculatoare și unități periferice de mare performanță, inteligență artificială și sisteme expert ;

- configurații de bază de natură să asigure capacități de prelucrare automată reale.

### **3. Câteva caracteristici ale producției de tehnică de calcul propuse pentru perioada 1986 — 1990**

Realizarea obiectivelor propuse pentru perioada 1986 — 1990 (ritmul fiind comparabil cu cel cunoscut pe plan mondial în domeniul tehnicii de calcul), implică creșterea substanțială a capacităților de producție de tehnică de calcul, preconizându-se realizarea investițiilor necesare la începutul perioadei.

Se subliniază că dimensionarea producției fizice a fost efectuată prevăzându-se producții importante în categoria mini și microcalculatoarelor și terminalelor universale și specializate.



#### 4. Cîteva considerații finale

Perioada actuală poate fi apreciată drept importantă pentru dezvoltarea producției de tehnică de calcul din țara noastră, care se găsește în faza de tranziție, spre compatibilizarea deplină cu cea a celorlalte țări, în primul rînd a țărilor socialiste, una din condițiile evidente pentru creșterea schimburilor reciproce de mărfuri și a intensificării colaborării și cooperării internaționale care impune, cel puțin din aceleași considerente, realizarea producției industriale de produse program și promovarea sistemelor orientate pe aplicații, atît cele generalizabile cît și cele la cheie.

Ne exprimăm părerea că dezvoltarea tehnicii de calcul conform direcțiilor propuse va avea contribuția așteptată la realizarea obiectivelor prioritare de creștere a productivității muncii și a eficienței activităților economice pe ansamblul economiei naționale.

#### ANEXA

##### Scurtă prezentare a anchetelor inițiate de I.T.C.I. la unitățile utilizatoare de tehnică de calcul

În vederea pregătirii elaborării proiectelor de dezvoltare a tehnicii de calcul la nivelul anului 1990 și a identificării problemelor pe termen scurt, I.T.C.I. a lansat două anchete : o anchetă cu caracter experimental la nivelul a circa 100 întreprinderi cu profil de electronică, electrotehnică și mașini unelte din MICM și o anchetă, la nivelul unui eșantion de 250 unități economico-sociale reprezentative pentru economia națională. În cele ce urmează se prezintă organizarea, conținutul, rezultatele și evaluarea celor două anchete, formulîndu-se cîteva propuneri.

1. *Ancheta la nivelul întreprinderilor cu profil de electronică, electrotehnică și mașini unelte.*

*Organizare :* proiectarea formularului ; completarea de verificare la Centrul de calcul I.T.C.I., definitivarea și multiplicarea ; lansarea prin poștă ; primirea răspunsurilor ; prelucrarea răspunsurilor (primite de la circa 57% din unități) ; verificarea prin sondaj la unități cu răspunsuri mai deosebite a corectitudinii acestora ; elaborarea raportului.

*Conținutul formularului :* 1. Identificare ; 2. Dotarea existentă (configurații) ; 3. Produse program utilizate în mod curent ; 4. Principalele limitări în utilizarea tehnicii de calcul în unitate ; 5. Măsuri apreciate ca esențiale pentru dezvoltarea tehnicii de calcul în cadrul ministerului ; 6. Opiniile privind serviciile BNP, respectiv ITCI ; 7. Contribuții așteptate din partea ITCI ; 8. Alte păreri și opinii.

##### *Principalele rezultate*

a) *Identificarea :* realizarea unei liste a delegaților unităților pentru problemele utilizării tehnicii de calcul în unitate.

b) *Dotare :* dotarea este în general redusă.

c) *Utilizarea echipamentelor :* cu excepții neesențiale toate unitățile care au răspuns dispun de cel puțin cîteva aplicații în exploatare curentă pe calculator Felix (propriu sau în colaborare cu Centrul de calcul teritorial sau al altor unități). Datorită insuficienței ca număr și tipuri de echipamente periferice și produse program, utilizarea mini și microcalculatoarelor este încă relativ redusă.



d) *Utilizarea tehnicii de calcul :*

Au rezultat următoarele concluzii :

- cu puține excepții, predomină aplicațiile de tip lot și sisteme informatice neintegrate; există totuși în exploatare și sisteme de conducere a producției de tip integrat (utilizând baze de date);

- configurația minimă a unui calculator/minicalculator destinat conducerii producției este 256KB, cu 3 unități disc 40 ÷ 50 MB, 2 ÷ 4 unități de bandă magnetică și cuploare necesare conectării câtorva zeci de terminale în special de tip industrial (inclusiv prin intermediul unor concentratoare). Actualele configurații standard cu care se livrează minicalculatoarele nu sînt optime pentru sistemele de conducere a producției;

- principalele preocupări ale unităților în domeniul utilizării tehnicii de calcul sînt (% răspunsuri) :

- realizare de baze de date, pregătirea tehnică a producției . . . 39
- programarea/lansarea/urmărirea producției . . . . . 26
- prelucrare interactivă/utilizare teletransmisie. (există o tendință de trecere la prelucrarea interactivă a informației) . . . . . 26
- asigurare condiții pentru dotare cu mini/microcalculator . . . 21
- aprovizionare/desfacere/gestiune stocuri . . . . . 19
- realizare soluție eficientă culegere/conversie date . . . . . 18

- domeniul prelucrării produselor/tehnologiilor cu suportul tehnicii de calcul este relativ slab reprezentat;

- în afară de configurație subdimensionată, utilizarea minicalculatoarelor în sisteme de conducere a producției este îngreunată și de lipsa unui sistem standard de gestiune a bazelor de date;

- anumite produse program din BNP (GESTOC, MIFIX, PERSO-RET) sînt frecvent utilizate;

- întreprinderile sînt interesate să facă schimb de produse program, să transmită soluțiile proprii, să colaboreze la elaborarea de noi sisteme generale.

e) *Principalele limitări în utilizarea tehnicii de calcul în întreprinderi sînt (% răspunsuri).*

- dotarea insuficientă cu echipamente de culegere a datelor de producție . . . . . 65

- dotarea insuficientă cu echipamente de culegere a datelor de evidență . . . . . 61

- numărul insuficient de suporturi magnetice . . . . . 61

- dotarea insuficientă cu micro și minicalculatoare . . . . . 53

- insuficiența pregătire informatică a utilizatorilor . . . . . 53

- lipsa produselor program satisfăcătoare pentru conducerea producției și analiza economică . . . . . 37

f) *Principalele păreri privind măsurile necesare în etapa actuală (% răspunsuri).*

- promovarea schimbului de informație tehnică . . . . . 60

- introducerea în sistemele de operare livrate de I.T.C.I. a unor produse program elaborate de unități și verificate în exploatare (inclusiv



dezvoltarea de produse program conform unui plan pus de acord cu întreprinderile)	51
● dezvoltarea de echipamente programabile pentru culegerea datelor și periferice	32
● dezvoltarea teleprelucrării, inclusiv realizarea unei rețele de calculatoare a MIEt și a legăturii calculator/minicalculator/microcalculator	12

#### *Evaluarea rezultatelor*

Proporția răspunsurilor întreprinderilor se poate aprecia ca satisfăcătoare, indicând interesul acestora în utilizarea tehnicii de calcul.

Concentrarea părerilor exprimate la întrebări deschise indică natura comună a problemelor la majoritatea unităților.

## *2. Anchetă la nivelul eșantionului de unități din economia națională.*

**Organizare:** proiectarea formularului; într-o formă adaptată prelucrării automate a informației; verificarea și definitivarea de comun acord I.T.C.I. și I.C.E. (unitatea beneficiară a contractului de cercetare); obținerea acordului ministerelor/organelor centrale cu privire la lansarea formularelor și eșantionul unităților; transmiterea formularelor; primirea; prelucrarea; elaborarea raportului de cercetare.

**Conținutul formularului:** 1. Identificare și caracteristici tehnico-economice generale; 2. Dotarea și experiența existentă în utilizarea tehnicii de calcul; 3. Utilizarea tehnicii de calcul pe domenii în prezent și în viitor; 4. Limitările existente în dezvoltarea tehnicii de calcul în unitate; 5. Necesarul de echipamente de tehnică de calcul în viitorul apropiat și cincinalul următor, inclusiv evaluarea importanței compatibilității; 6. Orientarea privind produsele program necesare; 7. Evaluarea colaborării cu BNP, ITCI, respectiv ICE; 8. Invitarea de a indica trei măsuri esențiale pentru promovarea producției și utilizării tehnicii de calcul în țara noastră.

#### *Principalele rezultate*

a) **Identificare:** realizarea unei liste a delegaților unităților pentru problemele utilizării tehnicii de calcul; experimentarea unei forme de organizare a unui registru al unităților.

b) **Dotarea cu echipamente de tehnică de calcul:** au rezultat următoarele tendințe:

Nr. crt.	Tip echipament	Nivel $\frac{1990}{1983}$ %
1.	Calculatoare	105
2.	Minicalculatoare	818
3.	Microcalculatoare	190
4.	Terminale	610
5.	Echipamente introducere/conversie date	702



c) *Utilizarea tehnicii de calcul*: au rezultat de exemplu următoarele tendințe :

Nr. crt.	Domeniu	Proporție utilizare ore calculator (%)			
		întreprinderi		alte unități	
		prezent	viitor	prezent	viitor
1.	Cercetare-proiectare produse/tehnologii	3	4	11	12
2.	Conducere proces tehnologic (inclusiv comandă numerică mașini unelte)	45	49	10	10
3.	Pregătire tehnică a producției (nivel bază de date a unității)	4	3	2	7
4.	Programarea/lansarea/urmărirea producției	14	12	11	19
5.	Analiză, calitate, întreținere, reparații, aprovizionare/desfacere/gestiune stocuri/transporturi, marketing	16	19	10	17
6.	Planificare tehnico-economică	1	2	3	4
7.	Personal, retribuire, finanțare, contabilitate	10	8	48	24
8.	Alte activități	7	3	5	7

d) *Necesarul de echipamente de tehnică de calcul*: (pe perioada 1985 — 1990). Efectuând o extrapolare pe categoriile de unități care a fost apreciate ca reprezentative (mai ales întreprinderi industriale) au rezultat de exemplu următorul necesar, comparativ cu calculele analitice de dimensionare efectuate de I.T.C.I.:

Nr. crt.	Tip echipament	Necesarul estimat conform anchelei
		Necesarul apreciat de ITCI
1	2	3
1.	Sisteme de introducere date	2,7
2.	Microcalculatoare	0,1
3.	Minicalculatoare	1,0
4.	Calculatoare de medie capacitate	3,4

Se constată că majoritatea opiniilor exprimate în etapa respectivă sînt puțin orientate pe prelucrarea distribuită, urmînd a se acționa în sensul creșterii semnificative în perioada următoare a ponderii prelucrării distribuite.

e) *Tendințe privind necesarul de produse program*. Tendințele privind necesarul de produse program sînt exemplificate în tabelul următor (nr. unități în %). (Au fost trecute numai frecvențele mai mari de 5 %).



Nr. crt.	Tipul produsului program (exemple)	Frecvența răspunsurilor privind produsele program necesare			
		imediat		în viitor	
		lot	interactiv	lot	interactiv
1.	Conducere proces tehnologic		15		23
2.	Comandă numerică mașini unelte	5	5		9
3.	Introducere/validare interactivă date		46		48
4.	Prelucrare geometrică date		17		21
5/6.	Gestiune rețele calculator/minicalcu- lator		28		38
7.	Programe generale creare/prelucrare fișiere	31	38	26	46
8.	Gestiune baze de date	12	42	20	57
9.	Programe matematice	11...15	13...21	10...16	14...23
10.	Programe cu caracter de pregătire, ana- liză și conducere a producției, ana- liză-calitate	16...26	13...31	12...22	17...36
11.	Programe cu caracter de gestiune și analiză economică	7...37	12...28	8...28	15...37
12.	Programare pentru prelucrarea evi- denței contabile/financiare statistice	39	23	28	34
13.	Sisteme de programe pentru proiec- tare produse/tehnologii	11	18	11	21

Se constată că un mare număr de unități consideră încă importante problemele de bază ale introducerii/validării interactive a datelor, ale creerii și prelucrării fișierelor, ale prelucrării evidenței. Este probabil că aceste răspunsuri sugerează de fapt necesitatea unor produse program standard și în același timp prietenoase, adaptabile condițiilor și necesităților utilizatorului.

f) *limitările privind utilizarea mai eficientă a tehnicii de calcul în unități.* Exemple de principale limitări semnalate au fost (% răspunsuri):

- dotarea insuficientă cu minicalculatoare . . . . . 55
- lipsa echipamentelor de culegere a datelor . . . . . 51
- insuficiente unități de memorie externă (cu discuri magnetice și cu bandă magnetică în configurație) . . . . . 45
- numărul insuficient de terminale (în configurații) . . . . . 44
- personal insuficient pentru cercetare/proiectare sisteme . . 39
- lipsă produse-program satisfăcătoare pentru gestiune baze de date . . . . . 32

g) *propuneri privind dezvoltarea domeniului tehnicii de calcul (întrebare deschisă)*

Cele mai frecvente propuneri formulate au fost (% unități):

- perfecționarea producției de tehnică de calcul în sensul: asigurării compatibilității cu produsele anterioare; asigurării de produse-program standard utilizabile de toate unitățile și livrabile o dată cu echipamentul; actualizarea centralizată a modificării produselor-program în funcție de modificările legislative; asigurării compatibilității fișierelor; asigurării compatibilității suporturilor de informație . . . . . 25
- creșterea fiabilității echipamentelor . . . . . 15
- dotarea unităților din economie conform unei concepții de sistem (care să asigure și compatibilitatea și generalizarea produselor-program de aplicație) . . . . . 10



● realizarea unei informări sistematice cu privire la producția și utilizarea tehnicii de calcul . . . . .	10
---	----

### *Evaluarea rezultatelor*

Ancheta la nivelul eșantionului de 250 unități a avut în mod inerent un caracter experimental, fiind recomandabilă efectuarea unei înregistrări cu caracter de masă (utilizînd formularul de anchetă existent, verificat în practică, cu unele simplificări de natura estimării unor întrebări).

Se poate aprecia că rezultatele sînt nereprezentative pentru categoria unităților mici similare întreprinderilor mici, unităților din agricultură și altor unități de interes local (în general lipsa unui registru statistic permanent oficial al unităților economico-sociale îngreunează organizarea de anchete valabile pentru ansamblul economiei naționale).

De asemenea, se poate aprecia că rezultatele concordă, în general, cu cele ale anchetei la nivelul întreprinderilor din MIET.

### *Valorificare*

Ancheta a constituit una din sursele calculului necesarului probabil în elaborarea proiectului de plan de dezvoltare a tehnicii de calcul.

### BIBLIOGRAFIE

- NICOLAE CEAUȘESCU, *Raportul Comitetului Central cu privire la activitatea Partidului Comunist Român în perioada dintre Congresul al XII-lea și Congresul al XIII-lea și activitatea de viitor a partidului în vederea îndeplinirii obiectivelor dezvoltării economico-sociale în cincinalul 1986 — 1990 și, în perspectivă, pînă în anul 2000, a României*, Edit. politică, București, 1984.
- \* \* \* *Directivile Congresului al XIII-lea al Partidului Comunist Român cu privire la dezvoltarea economico-socială a României în cincinalul 1986 — 1990 și orientările de perspectivă pînă în anul 2000*, Edit. politică, București, 1984.
1. M. DRĂGĂNESCU, *Realizări și perspective în informatică*, Sesiunea științifică Informatica și dezvoltarea economico-socială a României, ICI, 1983.
  2. V. BALTAC, *Optimizarea sistemelor de operare ale calculatoarelor numerice*, Edit. tehnică, București, 1974.
  3. V. BALTAC și colaboratorii, *Sisteme interactive, limbaje conversaționale*, Edit. tehnică, București, 1984.
  4. N. COSTAKE, M. MĂRȘANU, V. MĂNOIU, C. CĂLINESCU, C. VĂCEANU, *Proiect de program privind introducerea accelerată a automatizării și electronicii în economia națională, contribuția tehnicii de calcul*, Documentație cod PTK ITC, 1983.
  5. M. MĂRȘANU, V. MĂNOIU, I. DIAMANDI, C. CĂLINESCU, C. VĂCEANU, *Cercetări privind evoluția științei și tehnologiei în domeniul tehnicii de calcul*, Documentație ITC, 1983/1984.
  6. N. COSTAKE, M. MĂRȘANU, C. CĂLINESCU, C. VĂCEANU, *Ancheta efectuată la unitățile economico-sociale cu privire la problemele dezvoltării domeniului tehnicii de calcul*, Documentație cod ADTC ITC, 1983.



# CALCULATOARELE DE GENERAȚIA A CINCEA ȘI UNELE PROBLEME ALE CERCETĂRII-PROIECTĂRII ASISTATE DE CALCULATOR

GORUN MANOLESCU\*)

FIFTH-GENERATION COMPUTER SYSTEMS AND SOME PROBLEMS OF CAD SYSTEMS. The problems concerning CAD Systems will be analysed in the future. What has to be emphasized is the idea that Expert CAD Systems will be not capable to perform creative tasks. From this point of view, the author derives some aspects of creative reasoning (hierarchical and heterarchical structured reasoning, architectural reasoning, metaphorical reasoning etc.) which will be mean's concern.

## 1. Introducere

„Informația a devenit o materie primă care, deși nu figurează în nici un atlas economic, joacă în schimb, în societatea noastră, un rol mult mai important decât cel jucat de cărbune în secolul XIX în Anglia” [38]. Și dacă în alte domenii, mai ales în industrie (în fabricație), informația, împreună cu alte materii prime, se transformă în bunuri materiale obișnuite, în schimb, în cercetare-proiectare, „informația” se transformă tot în „informație”: studii, rapoarte tehnice, proiecte. Deci cercetarea-proiectarea este unul dintre domeniile în care, prin excelență, se utilizează drept materie primă „informația” și se produce, aproape exclusiv, „informație” (prelucrată).

Iată, în continuare, o definiție propusă pentru cercetare-proiectare cu ajutorul calculatorului (Cercetarea-Proiectarea Asistată de Calculator — CPAC) [39]: „CPAC este un ansamblu de tehnici informatice interactive care permit unui inginer să conceapă un produs, să efectueze calcule complexe cum ar fi: cele aferente structurilor de rezistență să obțină informații asupra componentelor standardizate și asupra proiectelor deja realizate, să transforme schițele în desene de execuție cotate, să execute verificări de coerență și toleranță, să obțină vederi în perspectivă, să ajute la elaborarea documentației tehnice, să realizeze modificări ale proiectului, inclusiv actualizarea documentației, să furnizeze date sub formă direct exploatabilă pe mașini unelte cu comandă numerică, să realizeze simularea funcționării prototipului viitorului produs tehnic”.

Conform cu definiția de mai sus, aproape că nu există activitate, subactivitate sau fază a procesului de cercetare-proiectare asupra căreia calculatoarele de generația a cincea să nu exercite un impact previzibil important, în măsura în care proiectul japonez se va finaliza, fie și numai într-o anumită măsură sau într-un interval mai îndelungat decât cel pre-

\*) Institutul de cercetare științifică și inginerie tehnologică pentru tehnică de calcul și informatică.



văzut inițial (până în 1990). În același timp, apariția conceptului de „calculator de generația a cincea” redeschide discuția într-o problemă controversată: triada „creativitate-inteligență-rutină” în procesul de cercetare-proiectare și diversele grade în care poate fi implicat calculatorul în fiecare element constitutiv al triadei. De aceea, paragraful următor va fi dedicat discutării aspectelor creative, inteligente, și de rutină ale procesului de cercetare-proiectare. În paragraful 3 vom analiza reconfigurarea interacțiunii „om-calculator” în CPAC prin prisma apariției calculatoarelor de generația a cincea. În fine, ultimul paragraf va fi destinat unor concluzii.

## 2. Triada „creativitate-inteligență-rutină” în procesul de cercetare-proiectare

Studierea procesului de cercetare-proiectare în ansamblu, pune în evidență încadrarea sa în categoria *sistemelor tehnologice* și, mai precis, în cea a *sistemelor tehnologice de prelucrare a informațiilor* [6].

De aici, se desprind următoarele:

- sistemele tehnologice, în general, sînt produse tehnice care trebuie să fie concepute și proiectate;
- procesul de concepere-proiectare, atunci cînd este implementat la nivel fizic, devine un sistem tehnologic, deci trebuie să fie el însuși conceput și proiectat. Ultima observație este esențială pentru scopul acestei lucrări.

Din studiile dedicate punerii în evidență a principalelor componente ale procesului [6, 10, 13] se relevă, aproape în unanimitate, existența următoarelor trei subprocese principale: (a) formarea așa numitului „concept” al noului produs; (b) proiectarea logic-funcțională; (c) proiectarea fizică sau de execuție.

Înregistrarea seacă, fără o analiză amănunțită, a existenței celor trei subprocese și, mai ales, a secvențialității lor, a provocat reacții îndreptățite [14]. Dar identificarea principalelor subprocese ale cercetării-proiectării oferă posibilitatea unei analize aprofundate a conținutului lor din diverse puncte de vedere. Unul dintre acestea este legat de ponderea diferită în care apar cei trei factori: creativitate, inteligență, rutină în cadrul fiecărui subproces.

În continuare vom discuta despre aspectele creative, inteligente și de rutină implicate în proces.

În conformitate cu studiile dedicate psihologiei creației [14, 25, 29], se observă că „procentul” de creativitate scade exponențial de la formarea „conceptului” spre proiectarea fizică, în timp ce „procentul” de rutină variază invers, iar cel de inteligență rămîne constant (vezi figura 1; pe această figură curbele ilustrează aspectele calitative; pentru date cantitative se poate consulta lucrarea [14]). De asemenea, trebuie precizat că, în cadrul psihologiei creației, se face o diferențiere netă între *creativitate* și *inteligență*.

O primă deosebire între aspectele creative și cele inteligente ale gândirii umane se poate face pe baza criteriului „tip de problemă” [25, 29]. Astfel, ar reveni aspectului creativ să se ocupe de problemele de tip *divergent* și celui inteligent, de problemele de tip *convergent*.



Tipul divergent include problemele care admit soluții multiple, fiecare soluție putînd fi obținută prin una sau mai multe căi de rezolvare. Rezolvarea acestui tip de probleme urmărește atingerea simultană a mai multor obiective care, de cele mai multe ori, nu pot fi precizate a priori, unele dintre ele conturîndu-se pe parcursul procesului de rezolvare. Acest tip de probleme este specific cercetării-proiectării, mai ales în cadrul formării „conceptului” viitorului produs tehnic.

Procent de

Creativitate (—)  
Rutina (---)  
Inteligenta (---)

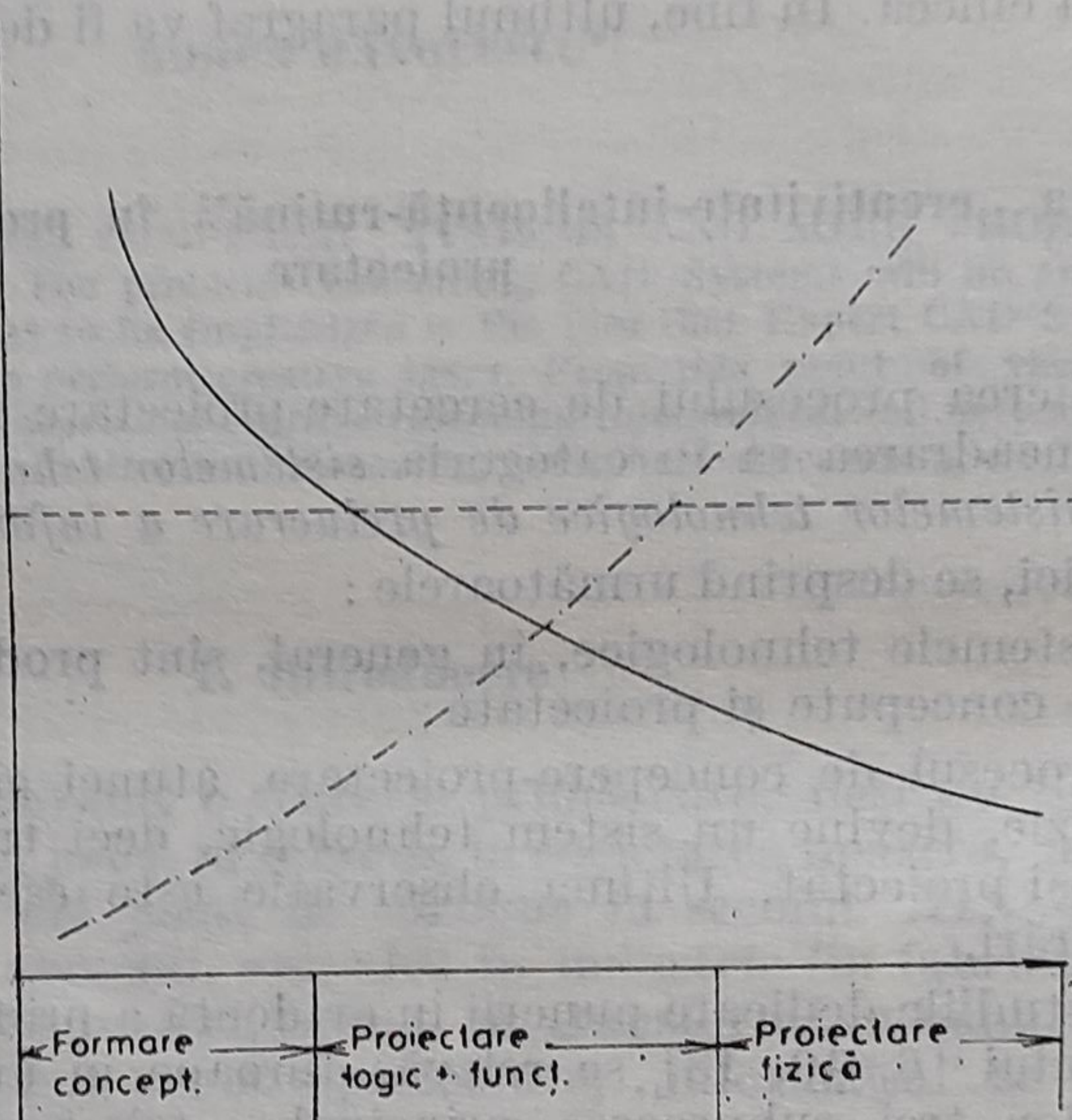


Fig. 1. — Aspecte ale gândirii implicate în cercetare-proiectare.

În opoziție, tipul convergent include problemele care admit o soluție unică, aceasta putînd fi obținută prin una sau mai multe căi de rezolvare. În această clasă de probleme intră, în primul rînd, problemele de care se ocupă matematica „pură”, inclusiv demonstrarea teoremelor. Tot aici își găsește locul și o serie de probleme ale matematicii aplicate cum ar fi cele de „programare matematică” deoarece, în acest caz, soluția optimă este unică (sau nu există) atîta timp cît s-au identificat corect restricțiile și funcția obiectiv.

O remarcă importantă : deși metodele euristice de rezolvare a problemelor sînt asociate, în literatură, cu aspectul creativ al gândirii, dacă privim lucrurile prin prisma „probleme de tip convergent în opoziție cu cele de tip divergent” se constată că, de fapt, în practică, euristica se aplică cu succes îndeosebi în rezolvarea neconvențională a unor probleme de tip convergent. Un exemplu tipic îl constituie demonstrarea teoremelor pe această cale [15]. Mai mult, dacă analizăm cu atenție cele trei genuri de transformări euristice propuse de Newell, Shaw și Simon [15], luate ca bază în teoria euristică a rezolvării problemelor, vom observa că fiecare dintre ele se reduce, în ultimă instanță, la determinarea diferențelor dintre un subscop, identificat în cursul procesului de rezolvare și scopul final, unic și precizat a priori și în reducerea acestor diferențe, pînă la anulare, în mod iterativ. Vom constata astfel, nu fără oarecare surprindere, că nu dispunem încă de un aparat teoretic adecvat pentru algoritmizarea



rezolvării euristice a problemelor divergente, aparatul teoretic existent fiind aplicabil numai în domeniul rezolvării problemelor de tip convergent.

Mai trebuie menționat faptul că testele de „creativitate” și „inteligentă” utilizate de psihologi [25, 29] au la bază disjuncția „divergent-convergent”. Interesant este și că s-a demonstrat statistic, pe baza unor asemenea teste, existența unui indice mediu de „inteligentă” la persoanele puternic creative și a unui indice scăzut de „creativitate” la persoanele cu „inteligentă” ridicată.

O a doua separare, mai fină, a aspectelor creative și inteligente și, de această dată și a celor de rutină, poate fi pusă în evidență dacă luăm în considerare etapele apariției și rezolvării unei probleme. Și anume: formularea problemei, algoritmizarea rezolvării ei și, în fine, rezolvarea efectivă (de observat analogia cu: formarea „conceptului”, proiectarea logic-funcțională și proiectarea fizică).

Păstrînd disjuncția „divergent-convergent”, aspectului creativ al gîndirii ar putea să-i revină [24]:

- rezolvarea directă, fără o algoritmizare prealabilă, a problemelor de tip convergent și/sau divergent, incomplet sau chiar greșit formulate; o astfel de rezolvare are la bază procese ale gîndirii, încă greu de explicat, de tip „intuitiv”;
- definirea de probleme divergente;
- definirea completă și/sau algoritmizarea unor probleme incomplet formulate de tip divergent;
- definirea incompletă (dar nu greșită) de probleme convergente.

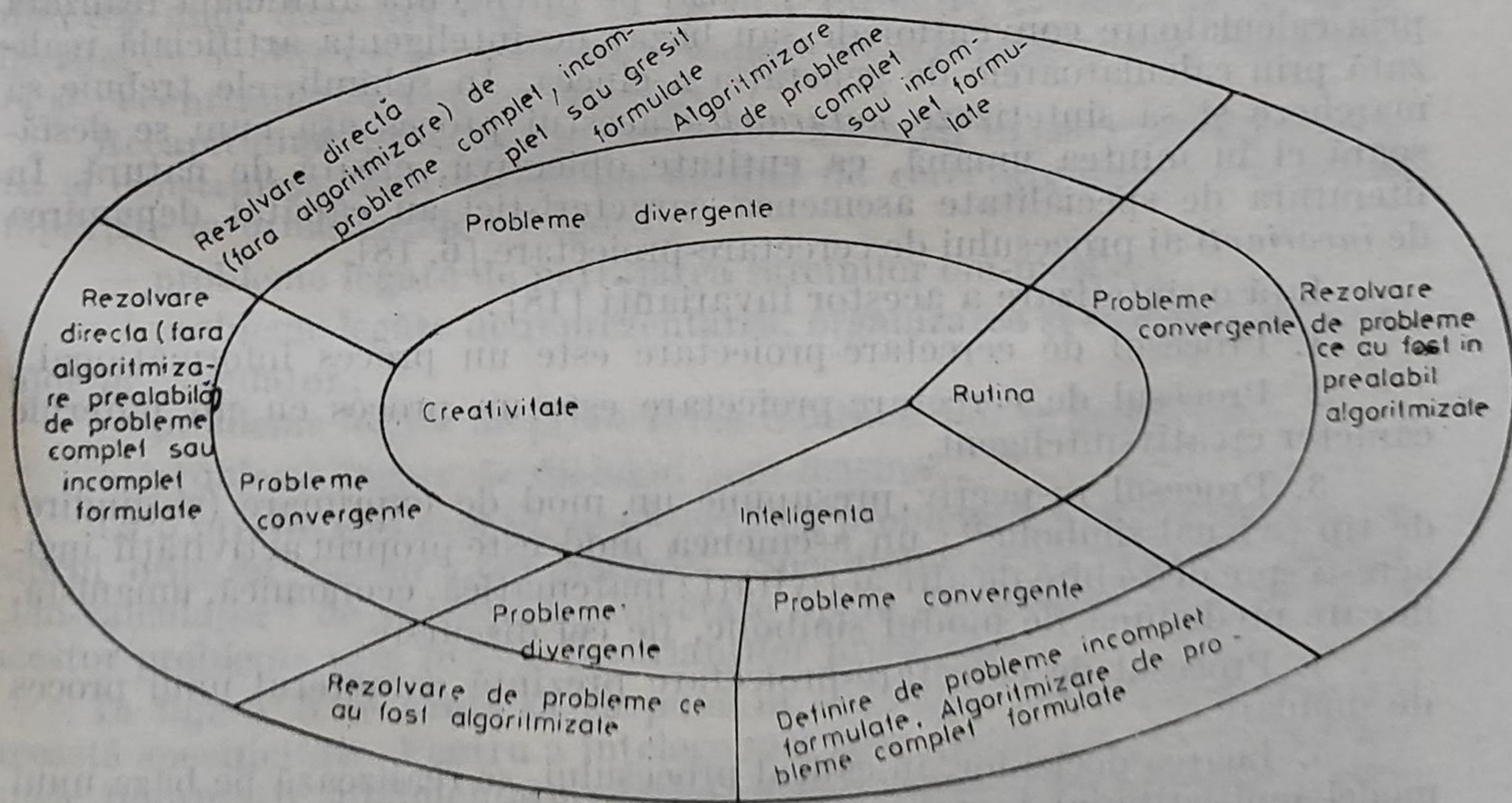


Fig. 2. — Sistematizarea aspectelor creative, inteligente și de rutină.

O explicație suplimentară în legătură cu rezolvarea problemelor greșit formulate. Rezolvarea unor asemenea pseudo-probleme apare atunci cînd, pe cale intuitivă, se ajunge la o soluție care se constată că, de fapt, aparține altei probleme decît celei avute în vedere. Exemple multiple de descoperiri „întîmplătoare”, dintre care cel al descoperirii penicilinei este cel mai cunoscut, sînt ilustrative pentru acest caz.



Aspectului inteligent al gândirii ar putea să-i revină [24]:

- rezolvarea efectivă de probleme divergente ce au fost, în prealabil, algoritmizate;

- definirea de probleme convergente;
- definirea completă și/sau algoritmizarea problemelor convergente incomplet (dar nu greșit) formulate.

În fine, aspectului de rutină al gândirii i-ar mai rămâne rezolvarea efectivă a problemelor convergente ce au fost, în prealabil, algoritmizate [14]. Este domeniul pentru care calculatoarele convenționale au fost create.

În figura 2 se sintetizează cele expuse în legătură cu clasificarea aspectelor creative, inteligente și de rutină ale gândirii umane.

### 3. Partajarea sarcinilor „om-mașină” în concepere-proiectare în contextul apariției calculatoarelor de generația a cincea

#### 3.1. Invariantii procesului de concepere-proiectare

Pentru a putea concepe și proiecta un sistem tehnologic (de prelucrare a informațiilor) destinat cercetării-proiectării de produse tehnice uzuale, este necesară punerea în evidență a caracteristicilor sale definitorii. Aceste caracteristici nu trebuie să fie influențate de modul particular — la nivel fizic — în care se implementează procesul: în sistem „om-mașină” cu mijloacele informaticii clasice, bazat pe inteligența artificială realizată prin calculatoare convenționale sau bazat pe inteligența artificială realizată prin calculatoarele de generația a cincea. În schimb, ele trebuie să marcheze și să sintetizeze *natura* acestui proces așa cum se desfășoară el în mintea umană, ca entitate obiectivă, creată de natură. În literatura de specialitate asemenea caracteristici au căpătat denumirea de *invarianti* ai procesului de cercetare-proiectare [6, 18].

Iată o sintetizare a acestor invarianti [18]:

1. Procesul de cercetare-proiectare este un proces informațional.
2. Procesul de cercetare-proiectare este un proces cu un puternic caracter creativ-inteligent.
3. Procesul respectiv presupune un mod de exprimare (și gândire) de tip „vizual-simbolic”; un asemenea mod este propriu activității ingineresti spre deosebire de alte activități: matematică, economică, umanistă, în care predomină fie modul simbolic, fie cel discursiv.
4. Procesul de cercetare-proiectare prezintă caracterul unui proces de simulare.
5. Luarea deciziilor, în cadrul procesului, se realizează pe baza unui model multicriterial în care obiectivele sînt adesea contradictorii și nu pot fi toate formulate apriori.
6. Procesul este caracterizat printr-o abordare de la general la particular, de la „concept” la desenul de execuție, de la ansamblu la detaliu, deci o abordare descendentă (top-down).
7. Pentru motivul că orice produs nou, oricît de simplu sau de complex ar apare după conceperea-proiectarea sa, înainte de a fi definit apare ca un obiect de complexitate ridicată, este necesar ca în timpul



procesului să fie posibilă o stăpânire a acestei complexități; în afara unei abordări top-down, o astfel de complexitate mai poate fi stăpinită și prin reluarea unor subprocesse implicate în proces.

Implementarea unui proces de concepere-proiectare în cadrul unui sistem „om-mașină” ridică o serie de probleme specifice ce vor fi discutate în continuare.

Faptul că procesul de concepere-proiectare este un proces pur informațional face posibilă, într-o măsură mult mai mare decât în cazul altor procese, utilizarea calculatorului.

Caracterul de simulare al procesului conduce la necesitatea utilizării la maximum a posibilităților puse la dispoziție de calculator în ceea ce privește simularea numerică, analogică și hibridă.

Caracterul creativ-inteligent implică o partajare a sarcinilor „om-mașină” într-o asemenea manieră încât entitatea „mașină” să-și poată realiza sarcinile atribuite la un nivel de competență comparabil cu cel uman și cu performanțe net superioare.

Modul de exprimare (și gândire) preponderent vizual-simbolic al procesului impune moduri specifice de reprezentare și stocare a informațiilor în calculator precum și moduri specifice de comunicare om-mașină.

Caracterul multicriterial, vizînd obiective adesea contradictorii, al modului de a lua decizii, presupune utilizarea la maximum a posibilităților calculatorului de a genera și analiza, cu performanțe net superioare omului, un număr mare de variante într-un timp redus.

Modul de abordare top-down și necesitatea reluării unor subprocesse sînt facilitate de utilizarea calculatorului în sensul că se poate crea posibilitatea unor descrieri din ce în ce mai rafinate, la intervale de timp diferite, cu modificări și verificări automate ale completitudinii, consistenței și necontradicției acestor descrieri.

Recapitulînd, principalele probleme care apar în cercetarea-proiectarea și construirea unui sistem om-mașină de cercetare-proiectare, pot fi împărțite în următoarele categorii:

- probleme legate de partajarea sarcinilor om-mașină;
- probleme legate de reprezentarea, organizarea și stocarea informațiilor în calculator;
- probleme legate de prelucrarea informațiilor în calculator;
- probleme legate de dialogul „om-mașină”.

La o privire mai atentă, se observă că problemele enumerate apar în cadrul oricărui sistem „om-mașină”. Specificitatea în cazul sistemelor „om-calculator” de concepere-proiectare apare din modul de rezolvare a acestor probleme prin prisma invariantilor procesului.

În figura 3 se caută să se pună în evidență, într-un mod sintetic, această specificitate. Pentru a înțelege mai bine figura 3 mai sînt necesare unele informații suplimentare. Astfel, baza (banca) de date (cunoștințe) tehnice, conține așa numitul „know-how” al unui domeniu particular de cercetare-proiectare (i.e. inginerie mecanică) și anume: standarde, informații despre materialele și semifabricatele existente și disponibile și caracteristicile acestora, proiecte deja executate în domeniu, informații asupra tehnologiilor necesare pentru realizarea ansamblelor, subansamblelor, reperelor etc. Această componentă a sistemului prezintă o dinamică relativ lentă a actualizărilor atît în ceea ce privește *volumul* cît și în ceea ce privește *structura* informațiilor. Baza de date (cunoștințe) a



proiectului este destinată să stocheze, în diversele perioade de timp ale procesului de concepere-proiectare, diferitele „imagini” ale noului produs, începînd de la „conceptul” acestuia și pînă la proiectul de execuție.

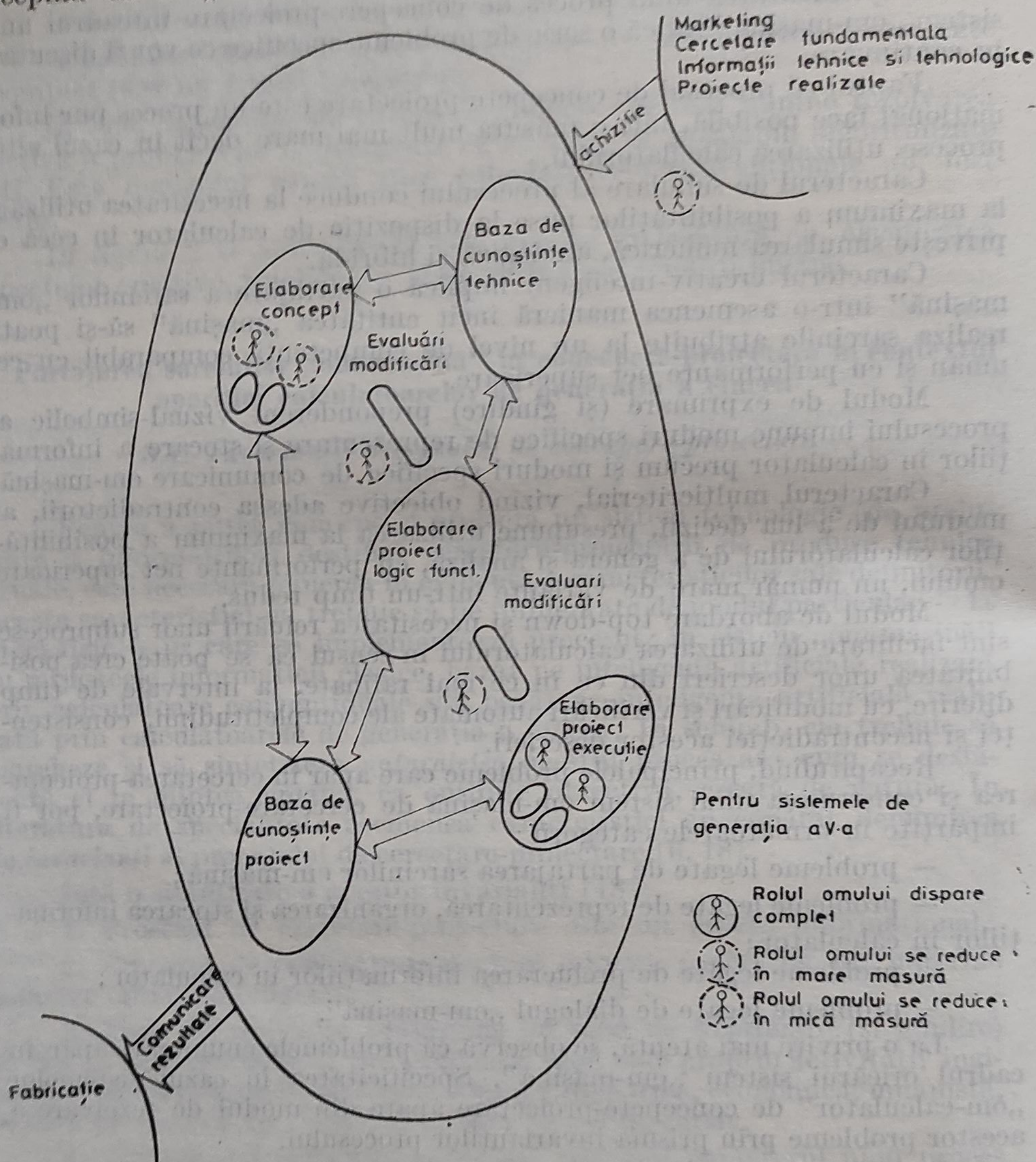


Fig. 3. — Sistem CPAC „om-mașină”.

Prin urmare, această componentă, în opoziție cu prima, va prezenta o dinamică mult mai accentuată a actualizărilor atât în privința volumului ei și în cea a structurării informațiilor.

### 3.2. Calculatoarele de generația a cincea și partajarea sarcinilor om-mașină în sistemele expert CPAC

Conceptul de „calculator de generația a cincea” reprezintă, de fapt, o sinteză a unor concepte, metode, tehnici, instrumente și realizări hard-



ware și software apărute în ultimii ani în informatică, mai ales în domeniul inteligenței artificiale. Dar tot așa cum un sistem nu se reduce la suma componentelor sale, nici conceptul de „calculator de generația a cincea” nu poate fi redus la elementele pe care se bazează și pe care le încorporează. Originalitatea noului concept constă tocmai în integrarea sistemică a acestor componente.

Deoarece proiectul japonez pentru calculatoarele de generația a cincea a fost pe larg expus în lucrarea lui A. Davidoviciu din prezenta eulegere de comunicări, nu vom mai insista prea mult asupra lui. Trebuie să spunem totuși că, după Moto-Oka [35]: „Calculatoarele de generația a cincea vor fi calculatoarele de procesare a cunoștințelor bazate pe teorii inovatoare și tehnologii care să ofere suportul necesar acestor teorii, depășindu-se astfel limitările calculatoarelor convenționale”.

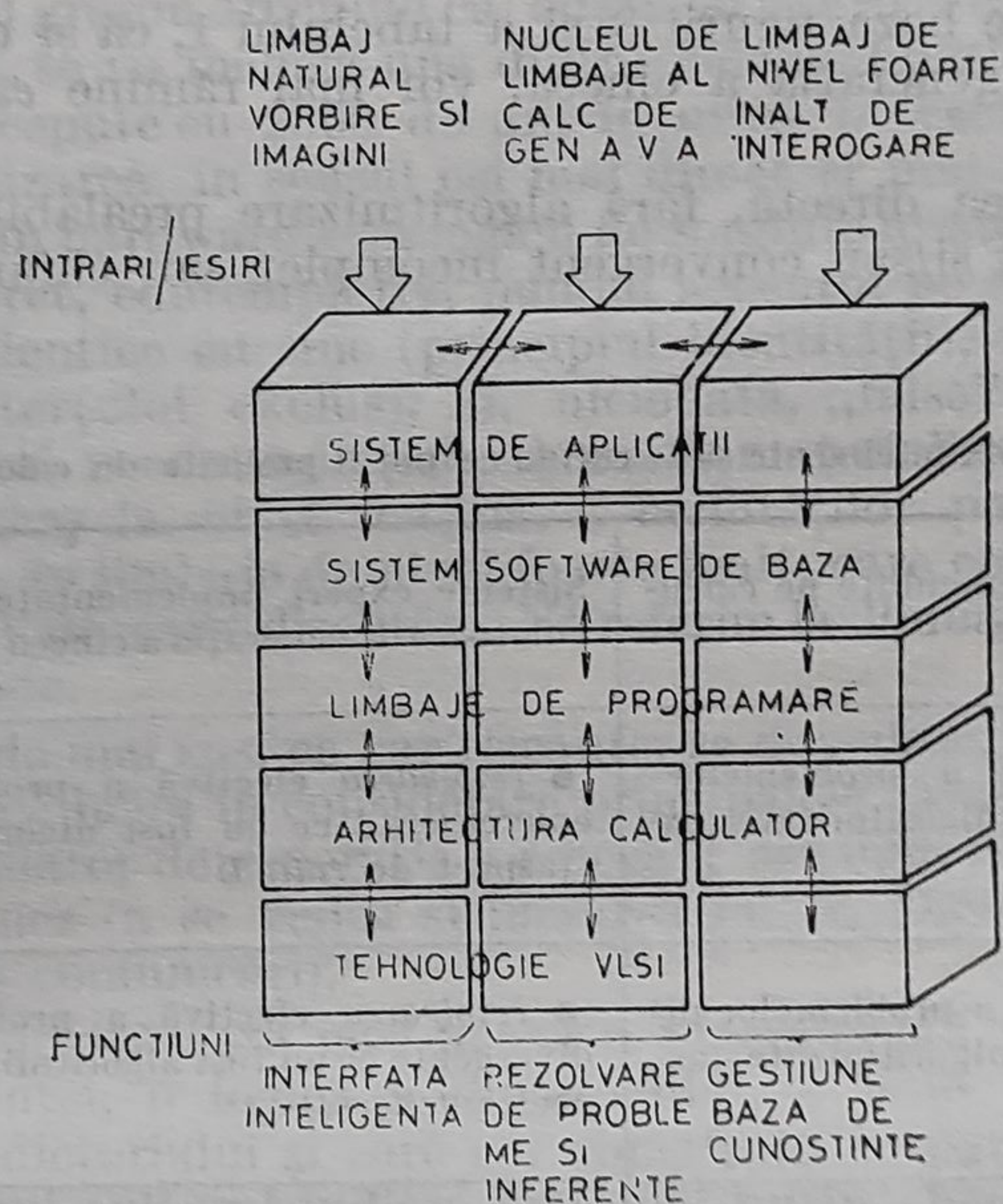


Fig. 4. — Capabilitățile și funcțiunile unui calculator universal de generația a cincea.

În figura 4 este redată o schiță a funcțiunilor și capabilităților unui calculator universal de generația a cincea, preluată din [35].

Din prezentarea [35] a principalelor caracteristici ale calculatoarelor de generația a cincea, se desprind două consecințe importante din punctul nostru de vedere și anume:

(a) principalele capabilități ale sistemelor expert (inteligente) implementate punctual astăzi pe calculatoarele convenționale la nivel software de ordin secund (utilizând hardware-ul și software-ul de bază al calculatoarelor convenționale) vor fi implementate în mod generalizat pe calculatoarele de generația a cincea la nivel hardware și software de ordin prim (sisteme de operare); acest lucru va conduce la o creștere spectaculoasă a performanțelor sistemelor expert;

(b) prin implementarea, la nivel hardware și software de bază, a unor facilități de comunicare om-mașină, neimplementate decît în mică



măsură, chiar la nivel software de ordin secund, pe calculatoarele convenționale, se vor lărgi, într-o măsură pe care poate nici nu o bănuim, capacitățile sistemelor expert (despre acest lucru vom mai vorbi în ultimul paragraf).

Ultima consecință, care reprezintă, de fapt, saltul calitativ pe care îl va înregistra apariția calculatoarelor de generația a cincea, ne oferă prilejul să prognozm o multiplicare a sarcinilor „mașină” pe care le vor putea prelua viitoarele sisteme expert CPAC.

Astfel, în tabelul 1, se prezintă o comparație între sarcinile pe care le poate prelua „mașina” în cadrul actualelor sisteme expert în general și CPAC în particular, și, respectiv, în cadrul acelorași sisteme implementate pe calculatoarele de generația a cincea. Același lucru am încercat să îl realizăm și pe figura 3.

Rezultă, pe baza figurii 3 și a tabelului 1, că și după apariția calculatoarelor de generația a cincea, vor mai rămâne *exclusiv* în sarcina omului :

• rezolvarea directă, fără algoritmizare prealabilă, a problemelor de tip divergent și/sau convergent incomplet sau chiar greșit formulate

Tabelul 1

Aspecte creative, inteligente și de rutină ce pot fi preluate de calculator în CPAC.

Sisteme expert implementate pe calculatoare convenționale	Sisteme expert implementate pe calculatoare de generația a cincea
• rezolvarea efectivă a problemelor convergente ce au fost algoritmizate (aspect de rutină)	• rezolvarea efectivă a problemelor convergente ce au fost algoritmizate (aspect de rutină)
• rezolvarea efectivă a problemelor divergente ce au fost algoritmizate (aspect inteligent)	• rezolvarea efectivă a problemelor divergente ce au fost algoritmizate (aspect inteligent)
• asistarea în definirea de probleme convergente incomplet (dar nu greșit) formulate (aspect inteligent)	• definirea de probleme convergente incomplet (dar nu greșit) formulate (aspect inteligent)
• asistarea în algoritmizarea de probleme convergente complet formulate (aspect inteligent)	• algoritmizarea de probleme convergente complet formulate (aspect inteligent)
• asistență în definirea de probleme convergente (aspect creativ)	• definirea de probleme convergente (aspect creativ)
	• algoritmizarea de probleme divergente complet formulate (aspect creativ)
	• asistență în definirea de probleme divergente (aspect creativ)



(asemenea facilități, puternic creative, sînt în mare măsură responsabile de apariția descoperirilor științifice, a invențiilor și inovațiilor și joacă un rol precumpănitor în formarea „conceptului” unui nou produs tehnic);

• definirea incompletă de probleme divergente.

Revenind la figura 1, se remarcă faptul că, în contextul apariției calculatoarelor de generația a cincea, aspectele puternic creative ale omului nu vor putea fi automatizate. Cel mult, în acest context, va apare o nouă partajare a sarcinilor „om-mașină” care să conducă la o eficiență sporită a sistemelor „om-calculator” în ansamblu.

#### 4. Încheiere

Credem că nu greșim afirmînd că, odată cu apariția calculatoarelor de generația a cincea, se va încheia una dintre cele mai pasionante aventuri ale cunoașterii, începute cu 2500 de ani în urmă în vechea Eladă. Este vorba de materializarea, în sensul cel mai direct și mai puțin metaforic, la nivel fizic — prin hardware — a logicii de sorginte aristotelică, cu caracter discursiv, discret, contemplativ, punînd accentul pe obiecte „văzute” drept imuabile, identice cu sine (principiul identității), false sau adevărate (principiul tertului exclus), și, niciodată, „false” și „adevărate” simultan (principiul necontradicției). Această materializare se va realiza prin implementarea, la nivel mașină, a facilităților pentru procesarea programelor scrise în limbaje de nivel foarte înalt, care constituie ultimile rafinări ale logicii de care discutăm. Ne referim la limbaje de gen KRL, LISP, PROLOG etc.

Afirmațiile de mai sus, ce par riscante, se dovedesc, la o privire mai atentă, firești, dacă luăm în considerare principalele caracteristici ale conceptului de „calculator de generația a cincea”, așa cum sînt ele prezentate în proiectul japonez (a se vedea și lucrarea lui A. Davidoviciu din prezenta culegere de comunicări).

Dar oare o logică care postulează o preeminență a actului asupra substanței și esenței, o logică a dinamicului, a unui real în devenire, o logică a contradictoriului și care s-a constituit, în paralel cu cea europeană, în contextul cultural indian, este mai puțin „logică”? Sau încă, o logică a „vizualului”, bazată pe o scriere ideografică în care un concept, cum ar fi, de exemplu, cel de „bătrîn”, are cel puțin 60 de nuanțe: „bătrîn cu baston”, „bătrîn peste 60 de ani”, „bătrîn sub 60 de ani”, „bătrîn în putere”, „bătrîn bolnav”, „bătrîn pe moarte” etc., logică cu o vechime, poate și mai mare decît cele apărute în contextul indo-european și care s-a format în spațiul cultural asiatic, este mai puțin „logică” decît primele două? Și, mai departe, logica „semnelor” și „simbolurilor”, pusă în evidență de C. Levi Strauss în cadrul culturilor primitive, nu este „logică” pentru că diferă de a noastră? Și întrebările de acest gen pot continua. Oare toate aceste logici, care corespund, parțial, cu logica aristotelică dar, în același timp, sînt și complementare ei, vor putea fi ele

\* De remarcă că, după mai bine de 2000 de ani, Europa reîntâlnește India, printre altele și în domeniul logicii, ceea ce a condus la apariția unor logici noi, care încep să se depărteze de cea convențională (aristotelică), prin relaxarea principiului identității și/sau a principiului contradicției (principiul tertului exclus fiind deja relaxat în cadrul logicilor polivalente și fuzzy). Ne referim la: logica deontică, logica epistemică, logica scopurilor, erotetica (logica întrebărilor) și nu, în ultimul rînd, la logica dinamică a contradictoriului (St. Lupașco).



„materializate” pe calculatoarele de generația a cincea întocmai ca cea de origine aristotelică? Cel puțin, bănuim că, o logică de natură vizuală, „ideografică”, proprie asiaticilor, va avea această soartă prin însăși natura lucrurilor.

Japonezii constituie una dintre națiunile asiatice care au reușit să înțeleagă gândirea europeană. Posedind acest avantaj, vor încerca ei — și vor reuși oare — ca, profitând de terenul comun al calculatoarelor de generația a cincea, să ne învețe o logică de tip asiatic? Dacă da, câștigul pentru noi va fi cert. Și aceasta deoarece, pe canal vizual, cantitatea de informație transmisă în unitatea de timp este de circa  $30 \div 40$  de ori mai mare decât cea transmisă pe canal auditiv. Vom redescoperi astfel una dintre multiplele posibilități de a gândi „în imagini”. Mai adăugăm că, o gândire de tip vizual, este proprie inginerilor și oamenilor de știință și că o astfel de gândire este strâns legată de mult discutata „intuiție”, reponsabilă de apariția descoperirilor, a invențiilor și inovațiilor care conduc, printre altele, și la proiectarea de noi produse tehnice. Vom ajunge, totodată, să putem depăși mai ușor barierele (lingvistice) pentru că „imaginea” prezintă un grad mai mare de universalitate decât vorbirea. Se va face, poate, un nou pas spre acel „limbaj universal” pe care îl visa Leibnitz sub forma unor structuri simbolice (matematice) și care s-au concretizat în actualele limbaje formale care, însă, nu și-au atins decât parțial scopul.

Și totuși, calculatoarele de generația a cincea vor continua să se bazeze, constructiv, pe „digital” care presupune existența unor atribute ca: discret, analitic, discursiv etc., caracteristici proprii emisferei stângi a creierului, neglijând încă aspectele continue, sintetizatoare, integratoare (gestaltice) care caracterizează emisfera dreaptă a creierului. Nu va genera acest fapt izbucnirea rapidă a unei noi „crize”, imediat după apariția calculatoarelor de generația a cincea? Lucrul este previzibil. Cum tot previzibil este și remediul: construirea unor noi calculatoare (de generația a șasea?) bazate pe microprocesoare hibride (digital-analogice), deja apărute în cadrul microelectronicii, microprocesoare ce vor oferi un suport fizic mult mai adecvat pentru realizarea unor procese de gândire mai apropiate de cele ale gândirii naturale. Vor fi posibile atunci implementări adecvate (și performante) ale unor noi tipuri de raționament, recent puse în evidență: ierarhic și heterarhic structurat, arhitectural, metaforic etc. raționamente responsabile, în mare măsură, de manifestarea plenară a capacităților umane creative. Astfel de raționamente sînt greu, dacă nu imposibil de realizat pe calculatoarele convenționale (chiar și prin simulare la nivel software). De asemenea ele vor fi, probabil, la fel de greu de implementat la nivel hardware pe calculatoarele de generația a cincea dar mai ușor de realizat la nivel software (cu performanțe însă scăzute).

Cum va arăta cercetarea-proiectarea asistată de calculator în spațiul cultural marcat de apariția calculatoarelor de generația a cincea și de cele ce vor urma? Greu de spus. Realitatea, (nu mai departe decât cea a zilelor noastre) se dovedește și s-a dovedit incomparabil mai plină de imaginație și fantezie decât previziunile. Un lucru însă pare extrem de probabil: noile capacități de comunicare om-calculator, mult mai apropiate de comunicarea naturală, capacități care au început să apară încă la nivelul calculatoarelor convenționale (calculatoare „user friendly”) și care vor căpăta o puternică dezvoltare așa cum prevede proiectul japonez, vor constitui unul dintre factorii dinamizatori ai procesului de apropiere din ce în ce mai puternică a gândirii artificiale de cea naturală.



Revenind acum la un context pragmatic imediat, extrem de apropiat nouă și „fierbinte”: ce trebuie să facă o țară care nu dispune de resurse (mai ales financiare) comparabile cu cele ale țărilor puternic industrializate, pentru a nu pierde pasul cu calculatoarele de generația a cincea și cu evenimentele următoare? Iată o întrebare deschisă spre nivelurile superioare de decizie. Considerăm totuși că nu greșim prea mult afirmând că încurajarea unor grupe de cercetare, mai ales în domeniul teoretic, grupe puternice calitativ și nu cantitativ, ar constitui unul dintre multiplele aspecte ce nu ar trebui neglijate. De asemenea, o modernizare curajoasă a programelor de învățământ, cel puțin în facultăți de profil (electronică, automatică-calculatoare) ar putea feri specialiștii de mâine de „șocul viitorului”.

#### BIBLIOGRAFIE

1. ALLAN, J. (ed.), *CAD Systems*, North Holland, 1977.
2. BOTEZATU, P., *Erotetica — logica întrebărilor*, Rev. de filosof., 4 (1980).
3. BOCHMANN, G. V., *Finite state description of communication protocols*, Comput. Netw., 2, Oct. (1978).
4. CARROLL, J. M., THOMAS, J. C., *Metaphor and the Cognitive Representation of Computing Systems*, IEEE Trans. on Man, Systems, and Cybernetics Society, March/Apr. (1982).
5. CULLINGFORD, R., E. et al., *Automated Explanations as a Computer Aided Design System*, IEEE Trans. on Man, Systems, and Cybernetics Society, March/Apr. (1982).
6. DAVID, B., *Methodologie pour la construction de systemes CAO*, These présentée a l'INP Grenoble, 1981.
7. DOUMEINGTS, G., *La production assistée par ordinateur*, Bull. de liaison de la recherche en informatique et automatique, 64 (1980).
8. FENCHEL, R. S., ESTRIN, G., *Self-Describing Systems Using Integral Help*, IEEE Trans. on Man, Systems, and Cybernetics Society, March/Apr. (1982).
9. GEORGESCU, I., *Componentele funcționale ale sistemelor expert*, ICI, 1982.
10. GRIFFITS, S., *Design methodologies — a comparison*, in: Marlow (ed.), IFOTEC State of the Art Report, Infotec Int. Lim., Breckshire, 1979.
11. HENDRIX, G., G., *Encoding knowledge in partitioned network*, in: Assoc. Netw., Acad. Press, New York, 1979.
12. HOCK, M. J., *Analysis of beginners' problem-solving strategies in programming*, in: Cognitive Engineering, Vrije Universiteit, Amsterdam, Aug., 1982.
13. JONES, J., *Design methods*, J. Willey, 1970.
14. KUDREAVTSEV, T., *Psihologia gândirii tehnice*, Edit. did. și ped., 1982.
15. LEDLEY, S. R., *Programarea și utilizarea calculatoarelor numerice*, Edit. tehnică, 1968.
16. MALHOTRA, A. et al., *Cognitive processes in design*, Int. Jour., of Man-Machine Stud., 2 (1980).
17. MANOLESCU, G., *Bazele teoretice ale sistemelor informatice: Probleme actuale și de perspectivă*, în *Inteligența artificială și robotica*, Edit. Academiei, 1983.
18. MANOLESCU, G. ș.a., *Sistem de instrumente informatice pentru realizarea aplicațiilor CPAC*, ICI, 1983.
19. MANOLESCU, G., TUDOR, A., *Definirea unui sistem om-mașină utilizând o gramatică bipartită*, BRI, 6 (1983).
20. MANOLESCU, G. et al., *A biparty grammar as a tool for defining a man-machine dialogue*, Computer Network, IFIP Conf., Sofia, Sep., 1984.
21. MARCUS, R. S., *User Assistance in Bibliographic Retrieval Networks through a Computer Intermediary*, IEEE Tran. on Man, Systems, and Cybernetics Society, March/Apr. (1982.)
22. PETERSON, J. L., *Petri Nets*, ACM Computer Surveys, 9, Sept. (1977).
23. POPA, C., *Logica epistemică și teoria cunoașterii*, Rev. de filosof., 4 (1980).
24. POPPER, K., *Logica cercetării*, Edit. științifică și enciclopedică, 1982.
25. POPESCU, NEVEANU, *Dicționar de psihologie*, Edit. Albatros, 1978.
26. PRICE, L. A., *THUMB: An Interactive Tool for Accesing and Maintaining Texts*, IEEE Trans. on Man, Systems, and Cybernetics Society, March/Apr. (1982).



27. ROBERTS, R., *HELP: A Question Answering System*, Proc. of Fall Joint. Conf., Arlington, 1978.
28. ROBERTSON, G. et al., *ZOG: A Machine Communication Philosophy*, Dept. of Comp. Sci., Carnegie-Mellon Univ., Pittsburgh PA, Aug., 1977.
29. ROȘCA, AL., *Creativitatea generală și specifică*, Edit. Academici, 1982.
30. SCHUBERT, L. et al., *The Structure and Organization of a Semantic Net for Comprehension and Inferences*, in: Assoc. Netw., Acad. Press, New York, 1979.
31. SHAPIRO, S. G., KWASNY, S. A., *Interactive Consulting via Natural Language*, Com. of ACM, 18, 5 (1975).
32. SHNEIDERMAN, B., *Multiparty Grammars and Related Features for Defining Interactive Systems*, IEEE Trans. on Man, Systems, and Cybernetics Society, March/Apr. (1982).
33. SONDHEIMER, N. K., RELLES, N., *Human Factors and User Assistance in Interactive Computing Systems: An Introduction*, IEEE Trans. on Man, Systems and Cybernetics Society, March/Apr. (1982).
34. SUSSMAN, G., *SLICES at the boundary between analysis and synthesis*, in: Assoc. Netw., Acad. Press, New York, 1979.
35. TRELEAVEN, P. C., LIMA, I. G., *Japan's Fifth-Generation Computer Systems*, Computer, Aug. (1982).
36. TUFİŞ, D., *Aspecte ale interacțiunii om-mașină prin intermediul limbajului natural*, in *Inteligența artificială și robotica*, Edit. Academiei, 1983.
37. WESLEY, M., *Design process*, Lecture Notes in Computer Science, nr. 89, Springer Verlag, 1980.
38. \* \* \* *L'informatique aujourd'hui*, Le Monde dossiers et documents, sept., 1982.
39. \* \* \* *La conception assistée par ordinateur*, Chaiier spécial nr. 49, 01 informatique n° 147 (1982).



# SISTEME SISTOLICE DE PRELUCRARE A DATELOR

ADRIAN PETRESCU\*)

SYSTOLIC DATA-PROCESSING SYSTEMS. The paper presents the main features of systolic systems implemented in VLSI. An example and some algorithms are also provided.

## 1. Introducere

Tehnologia circuitelor integrate pe scară foarte largă (VLSI) reprezintă unul din cele mai importante domenii de cercetare pentru calculatoarele din generația a cincea. În ceea ce privește producția circuitelor VLSI, două aspecte rețin în principal atenția cercetărilor.

Primul este legat de algoritmi VLSI, care se referă la funcțiile avute în vedere și modalitățile de implementare a acestora în VLSI [1]. Al doilea aspect are în vedere sistemele suport pentru proiectarea și fabricarea circuitelor VLSI, adică sistemele de proiectare asistată de calculator (SPAC).

Implementarea directă în VLSI a algoritmilor de mare complexitate oferă posibilitatea obținerii unor circuite specializate, realizate pe o singură pastilă de siliciu. Acestea pot fi utilizate în prezent ca dispozitive periferice specializate, conectate la un calculator convențional-gazdă sau pot fi folosite în calitate de componente, pentru sisteme mai specializate.

Algoritmi care sînt implementați eficient pe calculatoarele convenționale nu au în mod necesar implementări eficiente și în VLSI.

Pentru aceasta ei trebuie să posede o serie de proprietăți care să conducă la cîteva tipuri de celule simple, cu structuri, fluxuri de date și de comandă regulate, oferind posibilitatea interconectării lor într-o rețea uniformă.

Algoritmul trebuie să folosească în mod extensiv multiprelucrarea și principiul benzii de asamblare, pentru a asigura prelucrarea cu mare viteză a fluxurilor paralele de informații, pentru ca sistemul să ofere un răspuns în timp real.

Astfel, de algoritmi poartă numele de algoritmi sistolici [2], folosind un termen preluat din fiziologie. În structurile sistolice de prelucrarea a datelor, procesoarele joacă rolul unor inimi, care pompează (prelucrează) fluxuri multiple de date prin sistem, acesta din urmă avînd structura unei rețele.

Funcționarea ritmică, simultană a acestor procesoare paralele asigură un flux constant de date prin rețea și o mare productivitate.

În timp ce un procesor asigură transferul datelor spre alt procesor se realizează și prelucrarea acestora.

\*) Institutul politehnic din București.



La fiecare semnal de tact, generat de un orologiu electronic propriu, cu frecvența de ordinul zecilor de milioane de perioade secundă, se efectuează câte o operație de calcul, în fiecare procesor.

Rețelele unidimensionale de procesoare sistolice pot implementa algoritmi care se referă la probleme de înmulțiri de matrice și vectori, filtrare în timp real, transformata Fourier discretă, soluționarea sistemelor de ecuații liniare de formă triunghiulară, sortare în timp real, evaluare recursivă etc.

Rețelele pătrate sau hexagonale bidimensionale implementează în mod eficient algoritmi de identificare a caracterelor, operațiile în bazele de date relaționale, sortarea, interclasarea, prelucrări de imagini, înmulțiri de matrice etc.

Rețelele arborescente de procesoare sînt indicate pentru problemele de căutare, evaluarea paralelă de funcții, evaluări recursive etc.

Astfel, se apreciază că sistemele de calcul ale noii generații vor avea o structură descentralizată, bazată în mai mică măsură pe fluxul de comenzi și într-o măsură mai mare pe fluxul de date și pe reducere.

Aceste ultime două elemente vor asigura în principal diferențierea în raport cu calculatoarele convenționale, de tip von Neumann, bazate pe fluxul de comenzi.

În cadrul calculatoarelor convenționale se urmărește în mod explicit fluxul de comenzi, asigurat de execuția instrucțiunilor programului.

În calculatoarele orientate pe fluxul de date, disponibilitatea operanzilor declanșează execuția operațiilor asupra lor, în timp ce în calculatoarele bazate pe reducere, necesitatea unui rezultat lansează secvența de operații care va genera valoarea.

Microprocesoare conținînd peste 100 000 tranzistori pe pastila de siliciu au apărut deja de cîțiva ani.

Necesitățile tot mai mari în ceea ce privește puterea de calcul impun realizarea unor microprocesoare în tehnologie VLSI submicronică. În asemenea condiții apar întîrzieri importante pe traseele de legătură între elementele de calcul.

Gama procesoarelor VLSI are un spectru larg, mărginit la unul din capete de procesoarele specializate, iar la celălalt capăt de procesoarele universale. Spectrul va conține: pastile specializate, sisteme specializate, noi microprocesoare convenționale, mașini arborescente, calculatoare non-von Neumann.

Toate aceste procesoare specializate sau universale vor constitui blocuri constructive potențiale, pentru sistemele de calcul din generația a cincea. Ele se vor conforma unei noi arhitecturi și unei noi organizări a programelor.

## 2. Sisteme sistolice

Sistemele sistolice reprezintă o încercare de a cuprinde într-un cadru unic conceptele de paralelism, bandă de asamblare, structuri de interconectare, automate celulare și algoritmi.

La baza lor stau proprietățile unor circuite elementare cum sînt invertorul și tranzistorul de trecere, realizate în tehnologia NMOS [1]. Astfel, folosind invertoare (formate din tranzistoare cu canal indus și canal natural) interconectate prin tranzistoare de trecere, alimentate pe



grile cu fazele  $\phi_1$  și  $\phi_2$  ale unui orologiu, fără suprapunere, se poate obține un registru de deplasare (fig. 1).

În acest registru informațiile sînt stocate sub formă de sarcini pe grilele tranzistoarelor cu canal indus. Transferul informației se efectuează de la stînga spre dreapta. Pe durata fazei  $\phi_1$  se transferă informațiile de

Fig. 1. — Registru de deplasare dinamic pentru numere binare cu un rang.

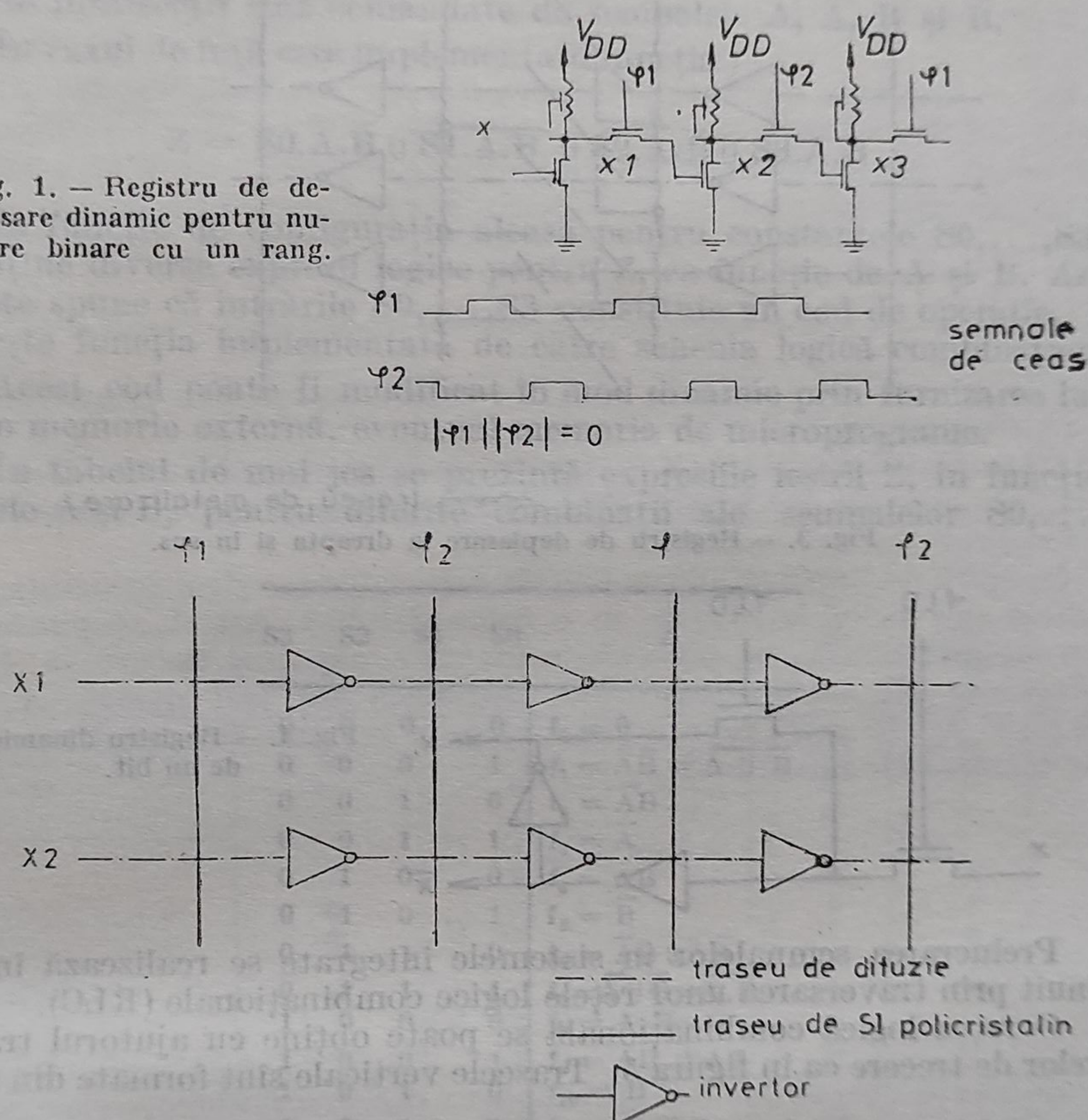


Fig. 2. — Registru de deplasare dinamic pentru numere binare cu mai multe ranguri.

la etajele invertoare impare, la cele pare, iar pe durata fazei  $\phi_2$ , de la etajele pare, la cele impare.

În mod asemănător se pot realiza registre de deplasare dinamice, cu mai multe ranguri (fig. 2), registre de deplasare la dreapta și în sus, în funcție de comenzile aplicate,  $\phi_2 \cdot SH$  și respectiv  $\phi_2 \cdot \overline{SH}$  (fig. 3).

Reprezentările din figurile 2 și 3 corespund diagramelor de linii [1] combinate cu simbolurile pentru invertoare. De remarcă faptul că la intersecția unui traseu de siliciu policristalin cu un traseu de difuzie se obține un tranzistor de trecere. Liniile orizontale reprezintă trasee de difuzie, iar liniile verticale constituie trasee de siliciu policristalin la care se aplică liniile verticale constituie grilele tranzistoarelor de trecere.



Registrul dinamic de un bit, din figura 4, se bucură de proprietatea că semnalul memorat se poate recircula pe durata semnalului activ  $\phi \cdot \overline{LD}$  sau se poate încărca pe durata semnalului activ  $\phi \cdot LD$ . La ieșire sînt disponibile atît semnalul direct, cît și cel negat.

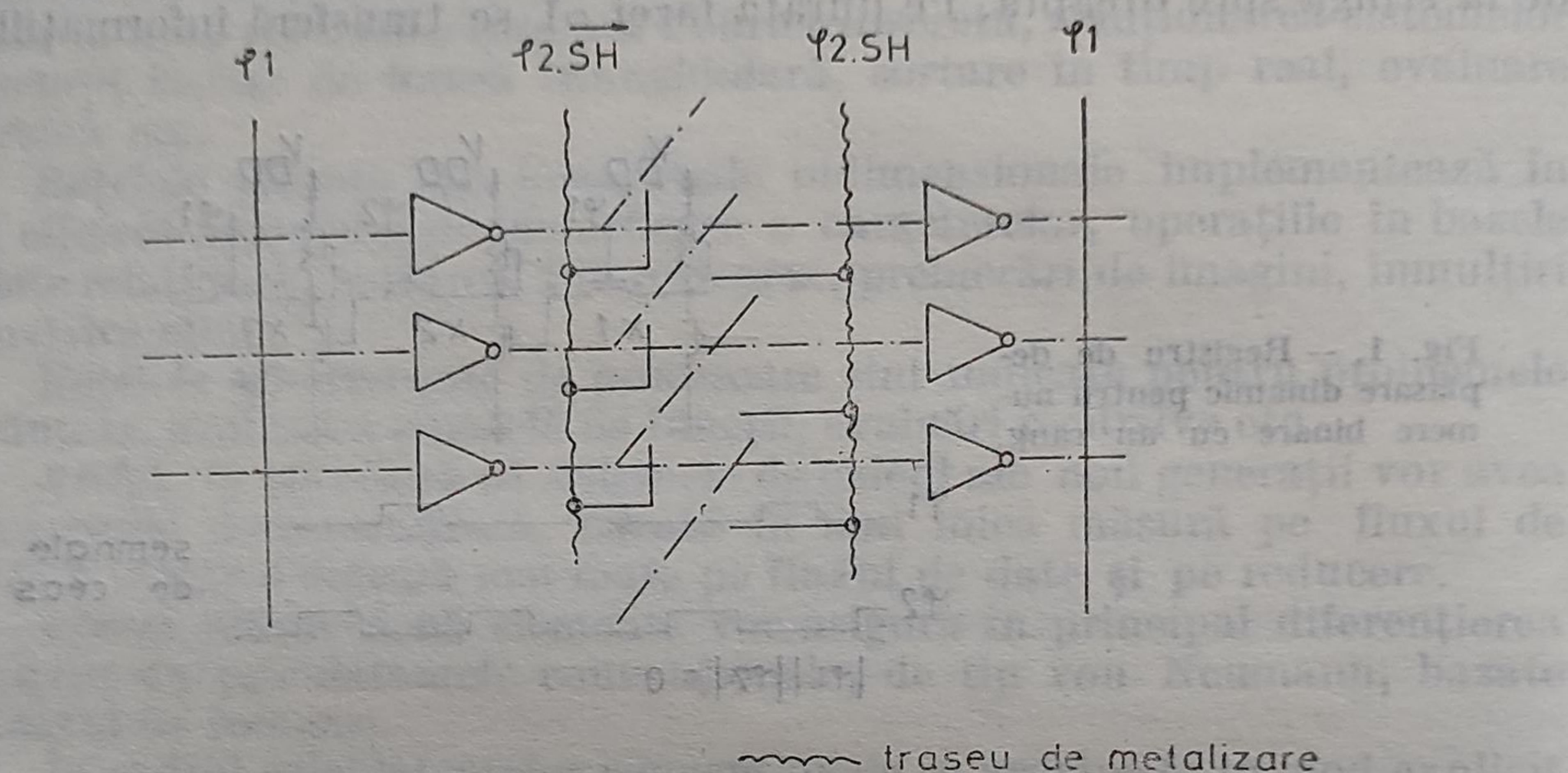


Fig. 3. — Registru de deplasare la dreapta și în sus.

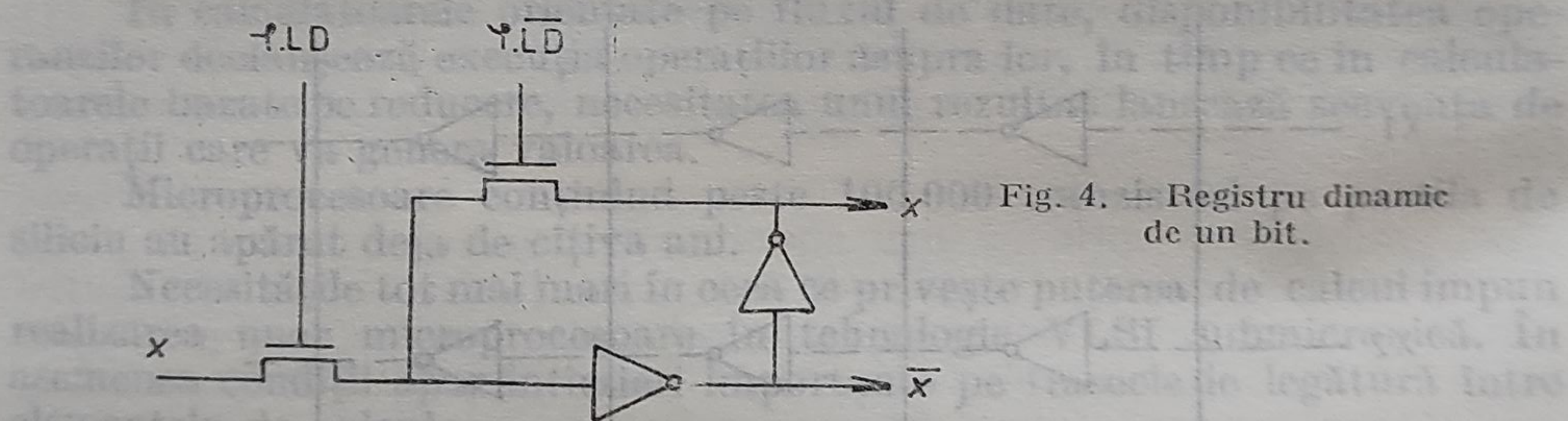


Fig. 4. — Registru dinamic de un bit.

Prelucrarea semnalelor în sistemele integrate se realizează în mod obișnuit prin traversarea unor rețele logice combinaționale (RLC).

O rețea logică combinațională se poate obține cu ajutorul tranzistoarelor de trecere ca în figura 5. Traseele verticale sînt formate din siliciu

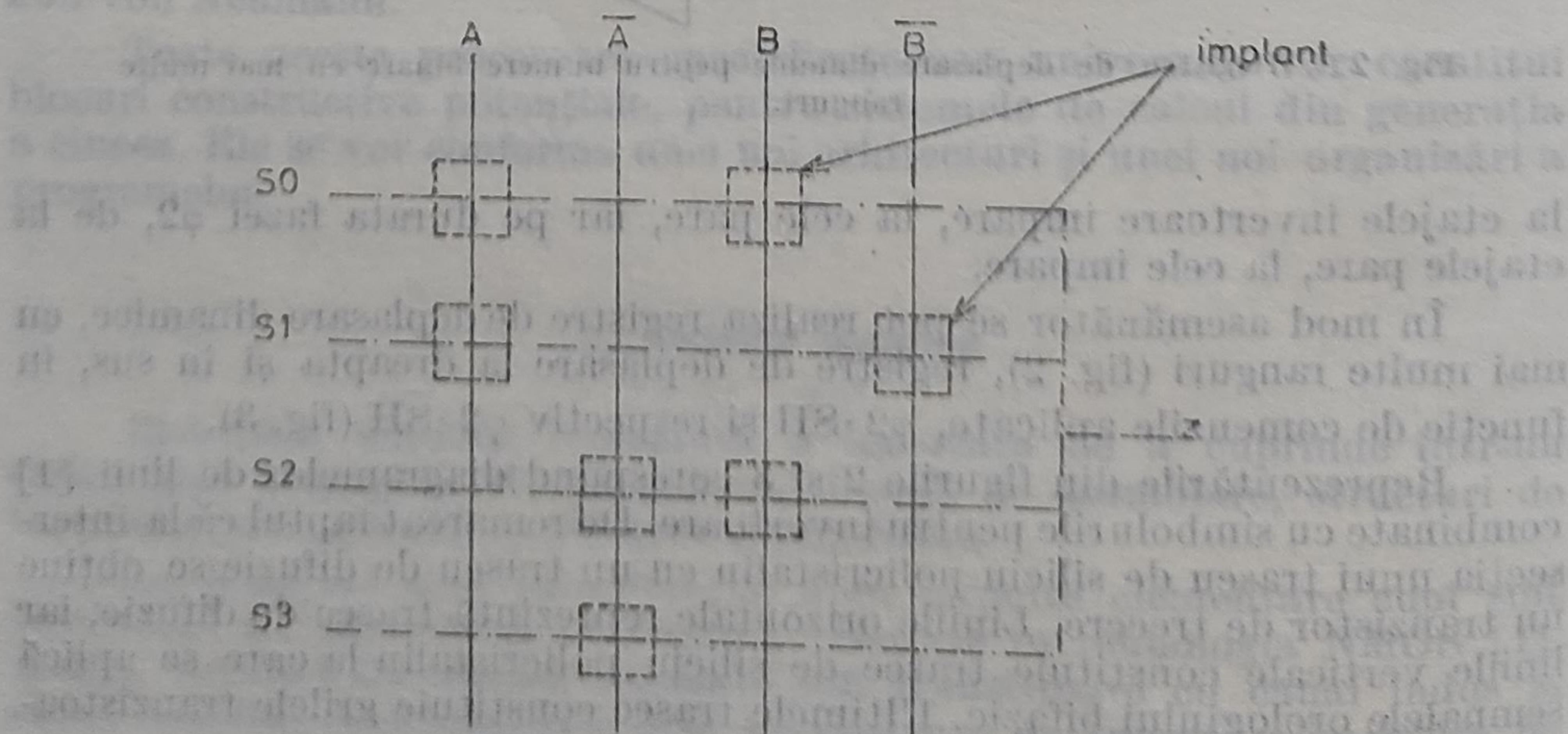


Fig. 5. — Rețea logică combinațională.



polieristalin. Ele sînt comandate de semnalele de selecție ( $A, \bar{A}, B, \bar{B}$ ). Traseele orizontale sînt de difuzie. La intrările din stînga se aplică constantele  $S_0, \dots, S_3$ . Ieșirea  $Z$  se obține în dreapta. La intersecțiile liniilor:  $S_0$  cu coloanele  $A$  și  $B$ ,  $S_1$  cu coloanele  $A$  și  $\bar{B}$ ,  $S_2$  cu coloanele  $\bar{A}$  și  $B$ ,  $S_3$  cu coloanele  $\bar{A}$  și  $\bar{B}$  s-au făcut implantări ionice pentru a obține tranzistoare cu canal natural, aflate permanent în stare de conducție. Celelalte intersecții sînt comandate de semnalele  $A, \bar{A}, B$  și  $\bar{B}$ .

În cazul de față este implementată funcția :

$$Z = S_0.\bar{A}.\bar{B} \cup S_1.\bar{A}.B. \cup S_2.\bar{A}.\bar{B} \cup S_3.A.B$$

În funcție de configurația aleasă pentru constantele  $S_0, \dots, S_3$  se pot obține diverse expresii logice pentru  $Z$ , ca funcție de  $A$  și  $B$ . Astfel, se poate spune că intrările  $S_0, \dots, S_3$  constituie un cod de operație, care stabilește funcția implementată de către schema logică combinațională.

Acest cod poate fi modificat în mod dinamic prin furnizarea lui de către o memorie externă, eventual memoria de microprograme.

În tabelul de mai jos se prezintă expresiile ieșirii  $Z$ , în funcție de intrările  $A$  și  $B$ , pentru diferite combinații ale semnalelor  $S_0, \dots, S_3$ .

$S_3$	$S_2$	$S_1$	$S_0$	$Z$
0	0	0	0	$f_0 = 0$
0	0	0	1	$f_1 = \bar{A}\bar{B} = \overline{A \cup B}$
0	0	1	0	$f_2 = \bar{A}B$
0	0	1	1	$f_3 = \bar{A}$
0	1	0	0	$f_4 = AB$
0	1	0	1	$f_5 = B$
0	1	1	0	$f_6 = \bar{A}B \cup A\bar{B}$
0	1	1	1	$f_7 = \bar{A} \cup \bar{B} = \overline{A.B}$
1	0	0	0	$f_8 = A.B$
1	0	0	1	$f_9 = \bar{A}\bar{B} \cup AB$
1	0	1	0	$f_{10} = B$
1	0	1	1	$f_{11} = \bar{A} \cup B$
1	1	0	0	$f_{12} = A$
1	1	0	1	$f_{13} = A \cup \bar{B}$
1	1	1	0	$f_{14} = A \cup B$
1	1	1	1	$f_{15} = 1$

O structură sistolică pentru prelucrarea datelor se poate realiza prin plasarea rețelelor logice combinaționale între registre de deplasare ca în figura 6. Codurile de operație  $COP_1, COP_2, \dots$  pot fi statice sau dinamice.

În ultimul caz, codurile de operație, corespunzătoare rețelelor logice combinaționale plasate la intrările registrelor sursă, trebuie forțate simultan cu faza orologiului pentru registrele sursă respective.

Un registru destinație asociat cu rețeaua logică combinațională de la intrarea sa poate fi asimilat cu un procesor. Astfel, sistemul constă dintr-o rețea de procesoare.



### 3. Exemplu de șir sistolic prioritar

În multe aplicații se impune inserarea de înregistrări într-o mulțime dată, precum și extragerea din acea mulțime a înregistrării cu cheia cea mai mică în conformitate cu o ordonare liniară.

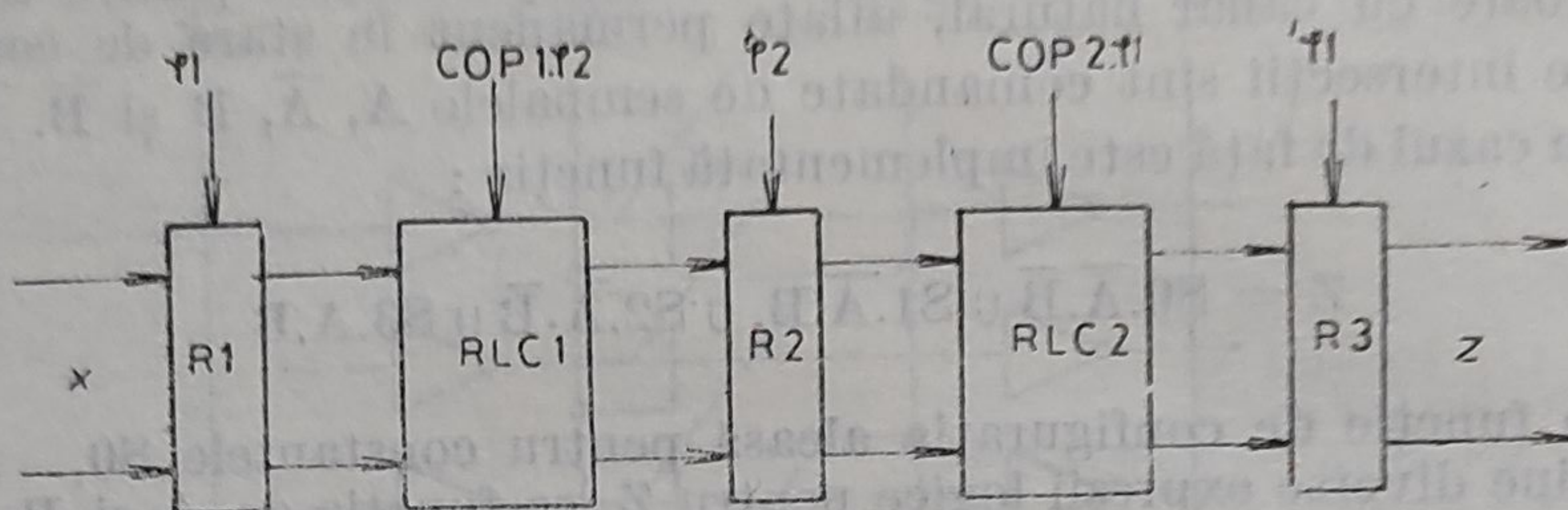


Fig. 6. — Structură sistolică.

O structură de date care permite asemenea operații poartă numele de șir prioritar [4].

Operația INSERTIE ( $S, i$ ) are ca efect înlocuirea șirului  $S$  cu șirul  $S \cup \{i\}$ , iar operația EXTRACȚIE MIN ( $S$ ) conduce la extragerea celui mai mic element  $i$  din  $S$ , înlocuind  $S$  cu  $S - \{i\}$ .

În afară de implementarea uzuală prin software șirul prioritar poate fi realizat sub formă hardware, ca o rețea sistolică. Astfel, se poate imagina o metodă bazată pe procesoare identice, capabile fiecare să sorteze trei elemente. Un asemenea procesor (sortator) are trei intrări:  $A, B, C$  și trei ieșiri  $A', B'$  și  $C'$ , reprezentând valorile minimă, intermediară și respectiv maximă ale intrărilor:

$$A' = \min(A, B, C),$$

$$B' = \text{med}(A, B, C),$$

$$C' = \max(A, B, C).$$

Ieșirile procesoarelor sunt prevăzute cu registre, iar logica este comandată cu semnale de ceas, astfel încât, în situația în care mai multe asemenea procesoare sunt interconectate, ieșirea unuia nu va interfera cu intrarea altuia.

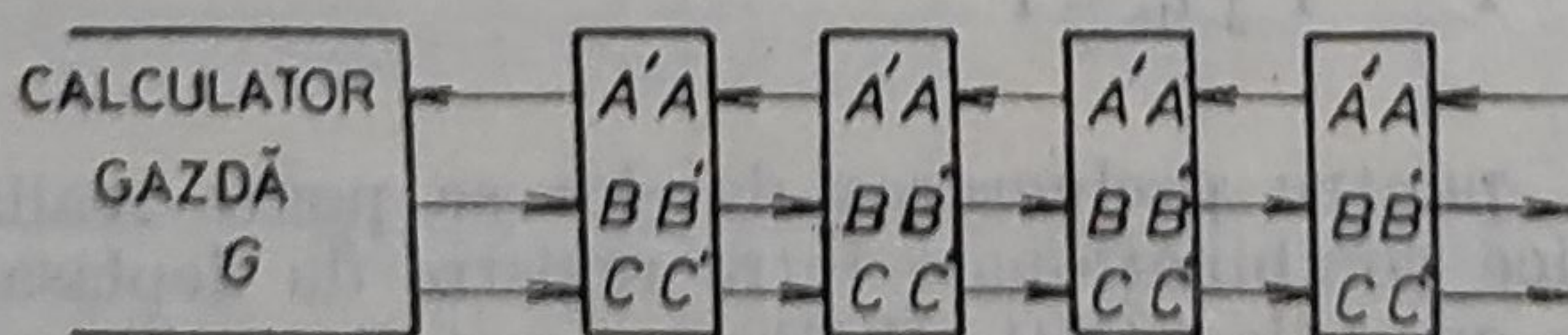


Fig. 7. — Procesoare interconectate într-un șir sistolic prioritar.

În figura 7 se prezintă modul în care se interconectează procesoarele pentru a realiza un șir sistolic prioritar. După cum se poate observa, valorile intermediară și maximă reprezintă ieșirile din dreapta, în timp ce ieșirea din stânga reprezintă valoarea minimă. Intrările  $B$  și  $C$  sunt furnizate de procesorul din stânga, iar intrarea  $A$  este asigurată de procesorul din dreapta. Astfel, cel mai mic element va fi menținut în stânga, iar cel



mai mare element, în dreapta. O cheie infinită,  $\infty$ , care este mai mare decât oricare altă cheie este furnizată în mod constant, ca o intrare în dreapta.

Celelalte două intrări din stînga sînt conectate la calculatorul gazdă. Inițial toate elementele din șir au cheia  $\infty$ . Pe măsură ce elementele sînt inserate în extrema stîngă, ele se vor deplasa spre dreapta înlocuind cheile  $\infty$ , care sînt eliminate în extrema dreaptă. Dacă vreodată ieșirile din dreapta nu sînt  $\infty$ , șirul și-a depășit capacitatea.

La primul impuls de ceas calculatorul gazdă G inserează la intrările procesorului P1 ( $G \rightarrow P1$ ) obiectele 7 și  $-\infty$ . Procesorul P1 sortează intrările 7 și  $-\infty$ , precum și intrarea  $-6$  furnizată de P2. La al doilea impuls de ceas transmite spre G, obiectul  $-\infty$  care este ignorat de către acesta.

OPERAȚIA	1	2	3	4	5	6
	$G \rightarrow P1$	$P1 \rightarrow P2$	$P2 \rightarrow P3$	$P3 \rightarrow P4$	$P4 \rightarrow P5$	$P5 \rightarrow P6$
INSERTIE 7	● 7	-6	●	$\infty$	●	$\infty$
	$-\infty$	●	-3	●	$\infty$	●
		●	5	●	$\infty$	●
	$-\infty$	●	-3	●	$\infty$	●
	●	-6	●	5	●	$\infty$
	●	7	●	$\infty$	●	$\infty$
EXTRACTIE MIN (-6)	●	-6	●	5	●	$\infty$
	$\infty$	●	-3	●	$\infty$	●
	$\infty$	●	7	●	$\infty$	●
	-6	●	-3	●	$\infty$	●
	●	$\infty$	●	5	●	$\infty$
	●	$\infty$	●	7	●	$\infty$
INSERTIE 4	●	-3	●	5	●	$\infty$
	4	●	$\infty$	●	6	●
	$-\infty$	●	$\infty$	●	$\infty$	●
	$-\infty$	●	5	●	7	●
	●	-3	●	$\infty$	●	$\infty$
	●	4	●	$\infty$	●	$\infty$
EXTRACTIE MIN (-3)	●	-3	●	7	●	$\infty$
	$\infty$	●	4	●	$\infty$	●
	$\infty$	●	5	●	$\infty$	●
	-3	●	4	●	$\infty$	●
	●	$\infty$	●	5	●	$\infty$
	●	$\infty$	●	7	●	$\infty$

La primul impuls de ceas vor funcționa procesoarele impare, iar la al doilea — procesoarele pare.

Calculatorul gazdă execută operațiile de INSERTIE și EXTRACTIE MIN. Modul de operare a șirului prioritar sistolic este oarecum diferit, deoarece el va executa, la fiecare impuls de ceas, o combinație de două operații de INSERTIE și o operație de EXTRACTIEMIN.



Pentru a executa inserția unui obiect  $i$ , cu ajutorul lui  $G$ , la intrările lui  $P1$ , în stînga, vor fi furnizate obiectele  $i$  și  $-\infty$ . La primul impuls de tact 1 este inserat  $i$ , iar  $-\infty$  este returnat către  $G$ .

Pentru a executa operația EXTRACTIEMIN, calculatorul  $G$  va furniza procesorului  $P1$  două obiecte  $\infty$ . După primul impuls de ceas,  $P1$  va returna obiectul cu cheia cu valoarea minimă. Cele două obiecte artificiale cu cheile  $\infty$  vor circula către dreapta, în șirul sistolic prioritar, eventual fiind eliminate la ieșirile din extrema dreaptă.

La fiecare pereche de impulsuri de ceas șirul prioritar este pregătit să execute alte operații de INSERTIE sau EXTRACTIEMIN.

Pentru a se plasa într-o rețea sistolică, un element poate avea nevoie de un timp relativ lung. O operație de inserție va necesita numai două impulsuri de tact. De asemenea, întrucît elementul minim din șir se află întotdeauna în stînga, o operație de EXTRACTIEMIN va necesita un interval constant de timp. Funcționarea rețelei sistolice este de tip banda de asamblare, astfel nu apare o degradare atunci cînd  $G$  execută mai multe cereri într-un șir prioritar, organizat pe una din liniile rețelei.

Se poate spune că o rețea sistolică are un răspuns în timp real față de operațiile INSERTIE și EXTRACTIEMIN, deoarece timpul de răspuns este constant în raport cu lungimea rețelei [5].

Structuri similare s-au evidențiat și în cazul sistemului sistolic destinat comparării șirurilor de caractere [6].

#### 4. Structura sistemelor sistolice

Un sistem sistolic reprezintă o rețea sincronă de procesoare paralele. Fiecare procesor din sistem este compus dintr-un număr constant de mașini secvențiale de tip Moore, care se bucură de proprietatea că ieșirile depind numai de stări. Sistemele semisistolice pot conține, atît automate de tip Moore, cît și automate de tip Mealy.

Aceste automate pot fi implementate în tehnologia VLSI sub forma rețelelor logice programabile (RLP), care au avantajul unei structuri regulate.

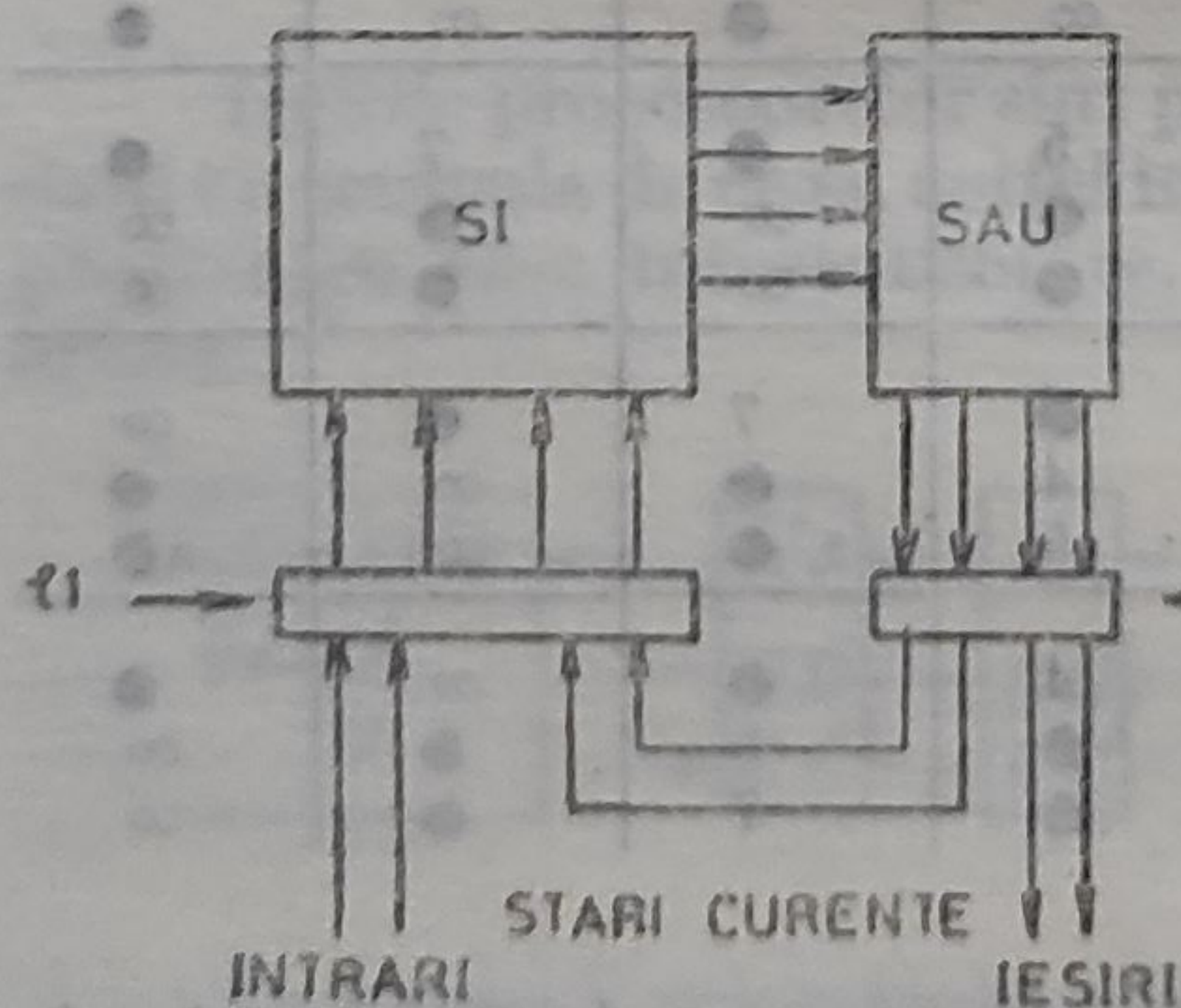


Fig. 8. — Automat de tip Moore.

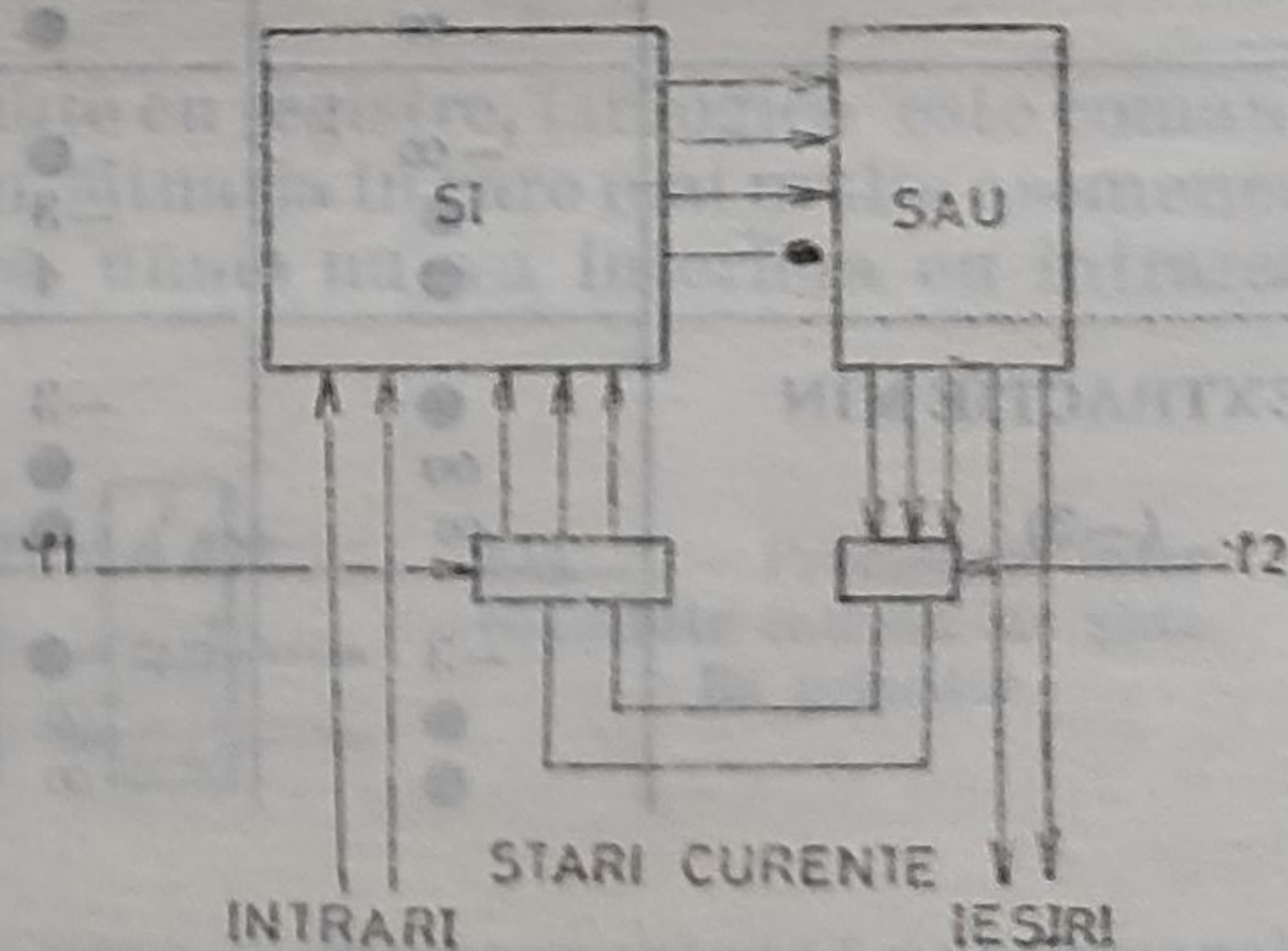


Fig. 9. — Automat de tip Mealy.

În figurile 8 și 9 sînt prezentate implementările în RLP ale automatelor de tip Moore și respectiv — Mealy. După cum se constată, datorită faptului că intrările și ieșirile din automatele Mealy nu sînt sincronizate cu fazele  $\phi 1$  sau  $\phi 2$ , aceste semnale se pot propaga prin circuitele combina-



ționale ale mai multor automate interconectate, ceea ce poate conduce la fenomene nedorite (curse, hazard etc). În sistemele sistolice se vor folosi automate Moore pentru ca perioada ceasului să nu crească odată cu dimensiunile sistemului, iar numărul de perioade de tact să fie o măsură a timpului, independentă de dimensiunile sistemului.

Structura unui sistem sistolic  $S(n)$  este dată de graful  $\mathcal{G} = (V, I)$  al celor  $n$  mașini interconectate, unde  $V$  reprezintă vîrfuri, iar arcele orientate  $I$  constituie interconexiunile mașinilor. Mașinile operează sincron folosind un ceas comun; timpul în sistem este măsurat prin numărul de cicluri ale ceasului. Toate mașinile din vîrfurile  $V$  sînt de tip Moore, în timp ce mașina gazda  $G$  poate fi tratată ca o mașină Turing, care asigură introducerea și extragerea informației în/din sistemul sistolic.

Pe baza grafului  $\mathcal{G}$  se poate defini pentru fiecare mașină o vecinătate  $v \in V$ , reprezentînd setul de mașini cu care ea comunică și care poate fi definită printr-un subgraf  $g(v, i)$ , unde  $i \in I$  constituie arcele corespunzătoare prin care se asigură comunicația directă.

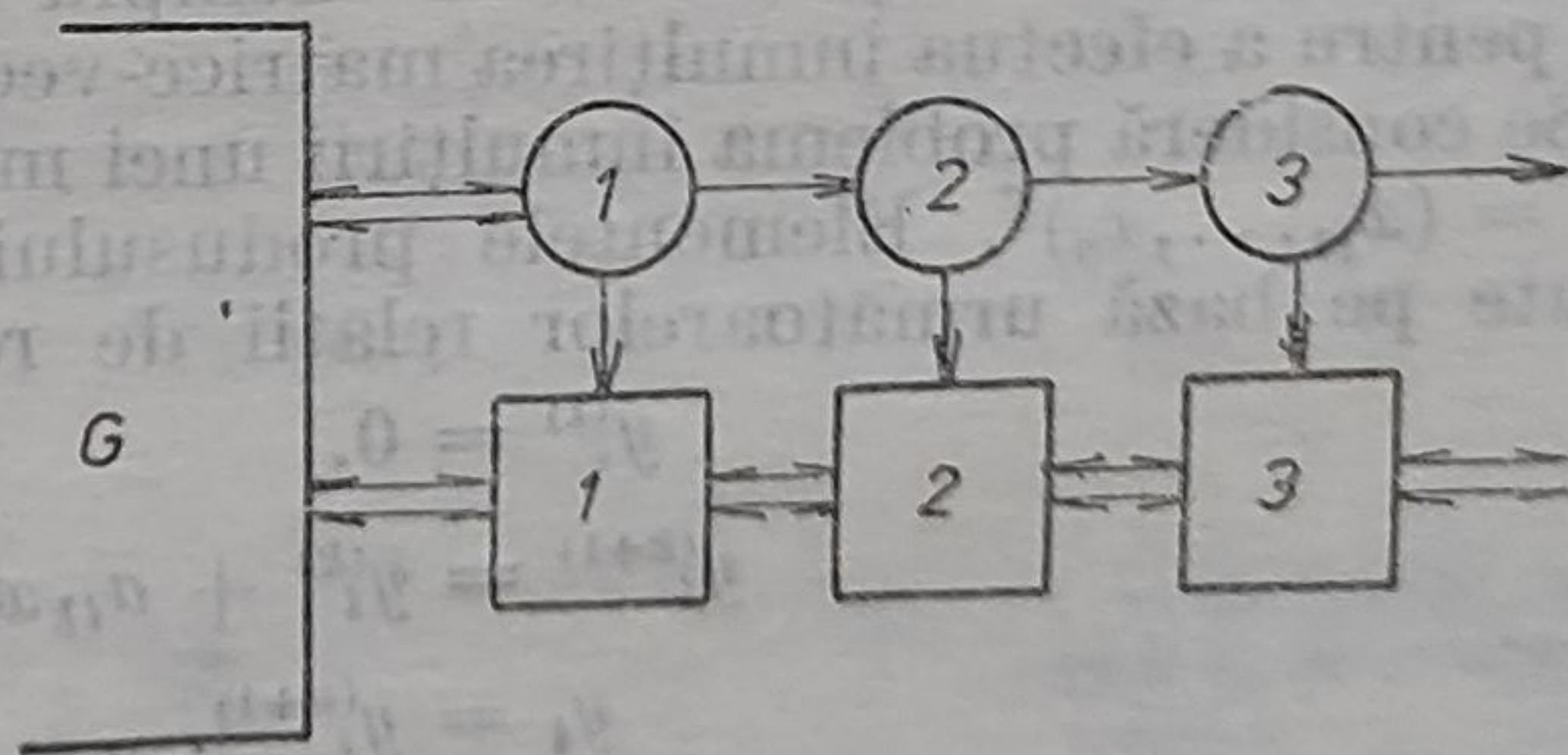
Pentru ca  $S(n)$  să fie sistolic este necesar ca mașinile Moore să fie „mici”, pentru a asigura corespondența, ca performanțe, între modelul sistolic și implementarea fizică. Sensul acestei restricții se leagă de existența unor constante  $k_1, k_2, k_3, k_4$  astfel încît, pentru toți  $n$  și toate  $v \in V - \{G\}$  să se îndeplinească condițiile ca :

- numărul stărilor mașinii  $|Q_v| \leq k_1$ ,
- numărul simbolurilor de intrare  $|X_v| \leq k_2$ ,
- numărul simbolurilor de ieșire  $|Z_v| \leq k_3$ ,
- numărul de vecini al mașinii  $|Vec(v)| \leq k_4$ .

Sistemele sistolice care au conexiuni numai cu vecinii cei mai apropiați sînt ușor de realizat în VLSI și asigură un timp de propagare pe interconexiuni mai mic decît timpul de prelucrare în procesoare.

În cazul în care se dorește difuzarea de la  $G$ , la celelalte mașini a unor informații simultan se pot folosi sisteme semisistolice, ca în figura 10.

Fig. 10. — Sistem semisistolic.



Propagarea se va face prin secțiunile combinaționale ale mașinilor Mealy; notate în desen cu cercuri. Fiecare mașină Mealy este conectată cu o mașină Moore (notată cu un patrat) și cu altă mașină Mealy. Comunicațiile între mașinile Moore și cu mașina gazdă  $G$  sînt notate cu linie dublă. În asemenea sisteme se pot implementa operațiile de difuzare de obiecte (DIFUZARE {i}).

În cazul cel mai simplu rețeaua de mașini Mealy poate fi o magistrală de comunicații.

În proiectarea sistemelor paralele mari este destul de dificil a se asigura sincronizarea, în sensul de a executa o operație dată la momentul corespunzător de timp.



Sistemele sistolice au dezavantajul că procesoarele pot comunica numai cu celelalte procesoare adiacente.

Datele trebuie să se propage prin rețeaua de interconectare astfel, procesoarele dispun de aceeași informație la momente diferite de timp. Comunicația globală, cu toate că reduce complexitatea sarcinilor de proiectare, implică comunicația pe distanțe mari și afectează performanțele. Astfel, se impune găsirea unor metode pentru implementarea sarcinilor comunicației globale numai prin comunicații locale.

## 5. Algoritmi implementabili în VLSI

Sistemele sistolice pot fi utilizate pentru implementarea eficientă a unor algoritmi importanți [7]. În cele ce urmează se enumeră mai multe exemple, pornind de la geometriile comunicațiilor:

1. Rețele liniare unidimensionale: înmulțiri matrice-vectori, filtrare în timp real, convoluție, transformare Fourier discretă, unități aritmetice de tip banda de asamblare, soluționarea sistemelor de ecuații liniare de formă triunghiulară, șiruri prioritare, sortare on-line, produs cartezian, evaluare recursivă.

2. Rețele patrute bidimensionale: identificarea de caractere, operații în bazele de date relaționale, transformata Fourier rapidă, sortare, interclasare, prelucrări de imagini.

3. Rețele hexagonale bidimensionale: înmulțirea matricelor, decompoziție LU prin eliminare Gaussiană fără pivotare, decompoziție QR, închideri tranzitive.

4. Rețele arborescente: algoritmi de căutare, probleme NP-complete, evaluări recursive, evaluări paralele de funcții, arbore sistolic de căutare.

5. Rețea de interschimb: transformata Fourier rapidă, sortare bitonică.

În continuare se prezintă un exemplu de rețea sistolică conectată liniar, pentru a efectua înmulțirea matrice-vector.

Se consideră problema înmulțirii unei matrice  $A = (a_{ij})$  cu un vector  $X = (x_1, \dots, x_n)^T$ . Elementele produsului  $Y = (y_1, \dots, y_n)^T$  pot fi calculate pe bază următoarelor relații de recurență:

$$\begin{aligned} y_i^{(1)} &= 0, \\ y_i^{(k+1)} &= y_i^{(k)} + a_{ik}x_k, \\ y_i &= y_i^{(n+1)}. \end{aligned}$$

Fie  $A$  o matrice banda  $n \times n$ , cu lățimea benzii  $w = p + q - 1$  și fie  $x$  un vector de lungime  $n$ . În cazul în care  $p = 2$  și  $q = 3$ , rezultă

$$q \left\{ \begin{array}{cccccc} a_{11} & a_{12} & & & 0 & \\ a_{21} & a_{22} & a_{23} & & & \\ a_{31} & a_{32} & a_{33} & a_{34} & & \\ & a_{42} & a_{43} & a_{44} & a_{45} & \\ & & a_{53} & & & \\ & & & 0 & & \\ & & & & A & \end{array} \right\} \begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_n \end{array} = \begin{array}{c} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \vdots \\ y_n \end{array}.$$



Produsul poate fi calculat prin plasarea elementelor lui  $X$  și  $Y$  prin rețeaua sistolică constând din  $w$  procesoare, care efectuează produse elementare. Funcționarea unui asemenea procesor este dată mai jos :

intrări			ieșiri		
$X$	$Y$	$Z$	$X'$	$Y'$	$Z'$
$x$	$y$	$z$	$x$	$y$	$z + xy$

Structura în termeni de intrări-ieșiri, pentru procesor, este dată în figura 11.

În figura 12 se prezintă structura sistolică la un moment dat. Funcționarea este următoarea : valorile  $x_i$ , care inițial sînt zero, se deplasează spre stînga, în timp ce valorile  $x_i$  se deplasează spre dreapta, iar  $a_{ij}$  se deplasează în jos. Fiecare  $y_i$  acumulează toți termenii săi  $a_{i,i-2}x_{i-2}$ ,  $a_{i,i-1}x_{i-1}$ ,  $a_{i,i}x_i$  și  $a_{i,i+1}x_{i+1}$  înainte de a părăsi rețeaua.

După  $w$  impulsuri de ceas, componentele lui  $y = AX$  se extrag la procesorul din stînga, cîte una la fiecare două unități de timp. Sistemul

Fig. 11. — Procesor elementar.

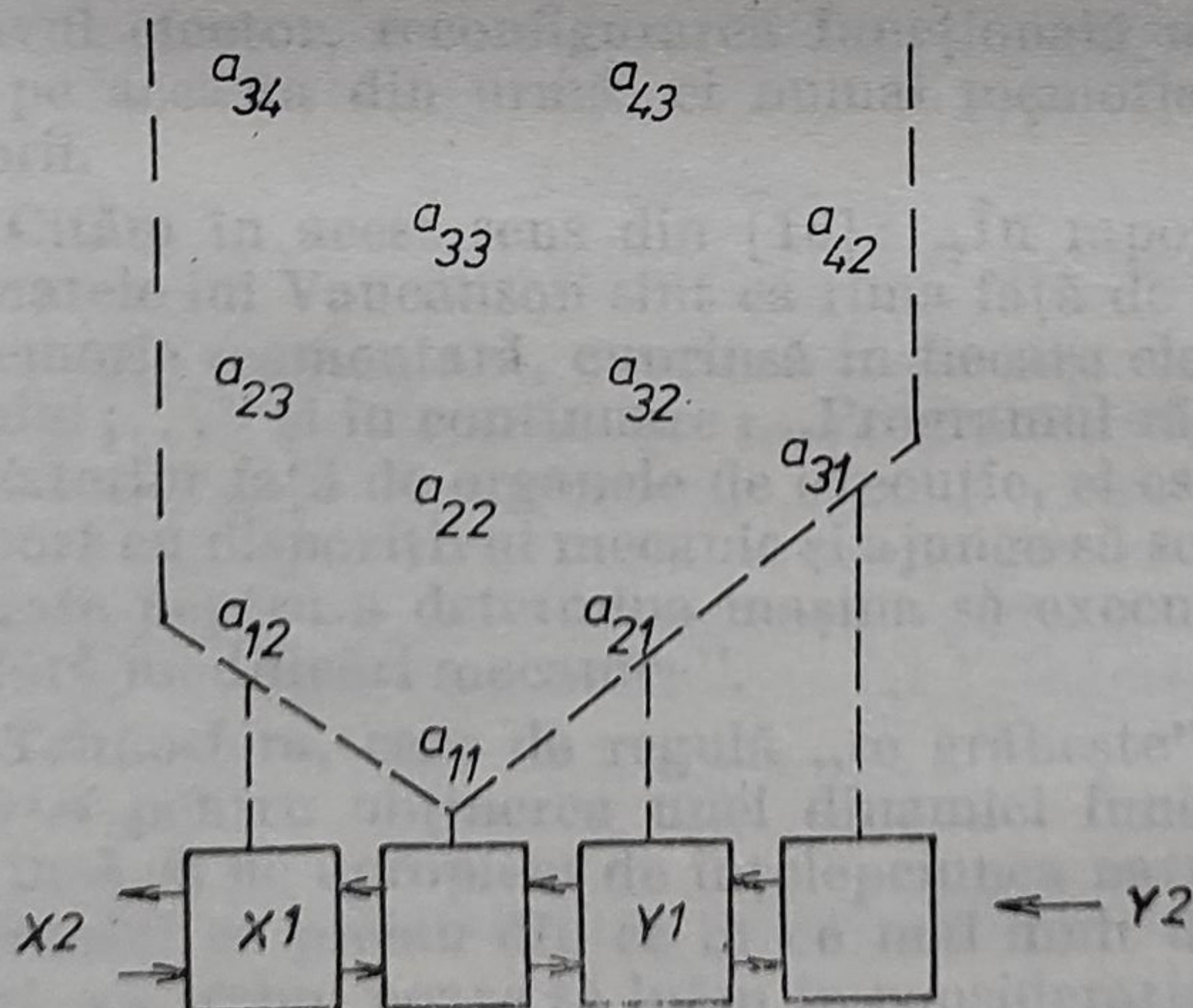
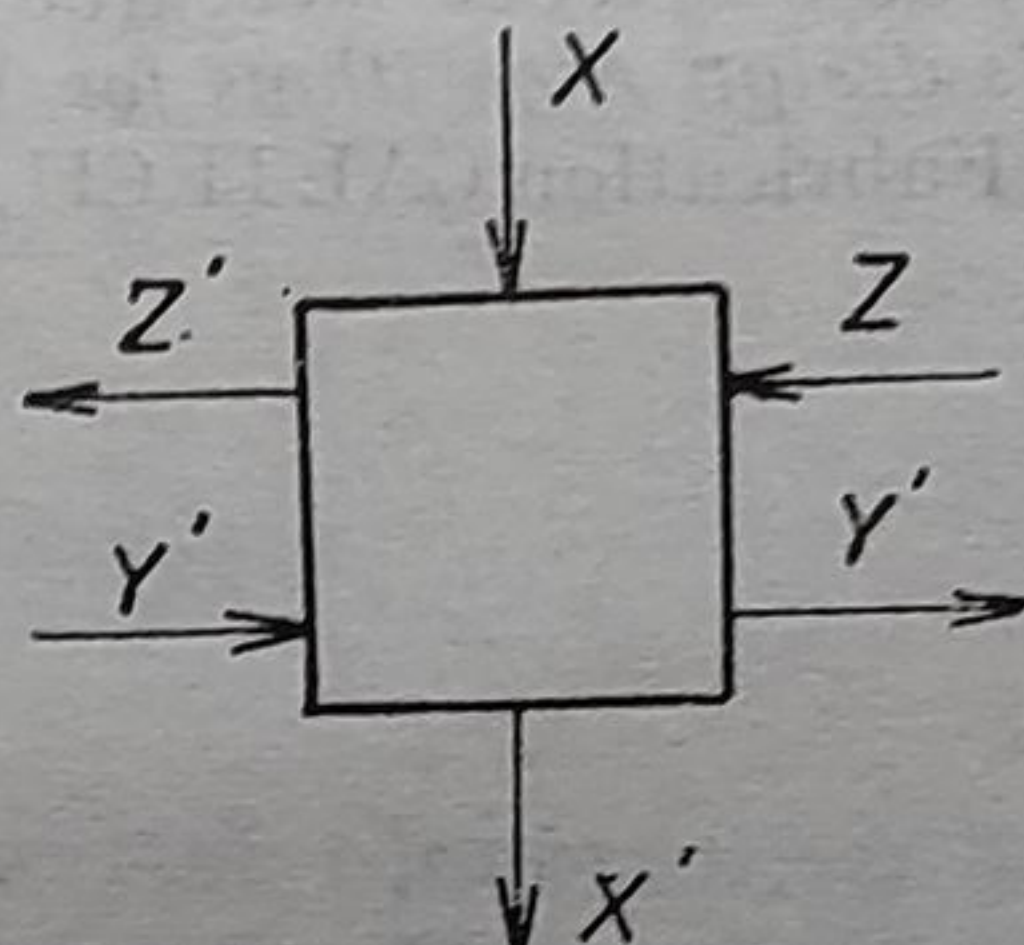


Fig. 12. — Structura sistolică la un moment dat.

sistolic va calcula toate cele  $n$  componente ale lui  $Y$  în  $2n + w$  unități de timp, în comparație cu timpul  $O(w n)$ , cerut de algoritmul secvențial, într-un sistem uniprocessor.



## 6. Concluzii

Ținând seama de implementările posibile în VLSI, ale unor structuri specifice de calcul și comunicații, elaborarea unor algoritmi eficienți, pentru a fi implementați direct în această tehnologie, prezintă o mare importanță.

În acest sens algoritmi sistolici reușesc să satisfacă dezideratele menționate mai sus.

## BIBLIOGRAFIE

1. C. MEAD, L. CONWAY, *Introduction to VLSI Systems*, Addison-Wesley Publishing Company, 1980.
2. H. T. KUNG, C. E. LEISERSON, *Systolic Arrays (for VLSI)*, In *Sparse Matrix Proceedings* 1978, p. 256—282, Society for Industrial and Applied Mathematics, 1979.
3. I. E. SUTHERLAND, D. OESTREICHER, *Microelectronics and computer science*, Scientific American, **237**, 3, Sept., 537—542 (1977).
4. A. V. AHO, J. E. HOPCROFT, J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Massachusetts, 1974.
5. C. E. LEISERSON, *Area-Efficient VLSI Computation*, Carnegie-Mellon University, 1981.
6. M. J. FOSTER, H. T. KUNG, *Design of special-purpose VLSI chips: exemple and opinios*, In *Computer Science Research Review*, Carnegie-Mellon University, 1978—1979.
7. H. T. KUNG, *Let's design Algorithms for VLSI Systems*, Conference on VLSI: Architecture, Design, Fabrication CALTECH, 1979.



# COMPATIBILIZAREA FUNCȚIE-STRUCTURĂ CA MECANISM AL EVOLUȚIEI ARHITECTURALE

GHEORGHE M. ȘTEFAN\*), AUREL PĂUN \*)

„Hardware engineers are concept-limited,  
not component-limited”

E. A. FEIGENBAUM, 1982

FUNCTION-STRUCTURE MERGING AS A MECHANISM OF ARCHITECTURE EVOLUTION. The gap between function and structure was produced in digital systems by means of information. Firstly the paper defines information and its materialization in digital machines as well as class relations between language hierarchies and physical structures. The evolution of machine architectures is emphasized starting with arithmetic through logic to knowledge-based ones. Finally the authors present a personal viewpoint regarding the role of algorithms in new computing generation machines.

## 1. Introducere

Mult înainte ca automatele să devină electronice s-a manifestat o polarizare extremă a două categorii de soluții. Una presupunea dispozitive în care memoria nu se integra ca o componentă distinctă, chiar dacă exista și determina funcția ansamblului, iar alta în care memoria se izola de dispozitivul efector, reconfigurarea funcțională neafectînd, în limite foarte largi pe aceasta din urmă, ci numai memoria exteriorizată elementelor efectorii.

Cităm în acest sens din [16]: „În raport cu mașinile electronice, automatele lui Vaucanson sînt ca rîma față de mamifere, adică organisme cu memorie segmentară, cuprinsă în fiecare element component al dispozitivului;...” și în continuare: „Programul războiului de țesut Jacquard este exterior față de organele de execuție, el este, dacă vrem „inteligent” în raport cu dispozitivul mecanic și ajunge să se schimbe banda de cartele perforate pentru a determina mașina să execute o serie de operații diferite, fără modificări mecanice”.

Tehnosfera, care de regulă „se grăbește”, a stimulat exteriorizarea memoriei pentru obținerea unei dinamici funcționale foarte mari. Dacă vrem însă să ne apropiem de înțelepciunea naturii, în tendința de a concepe mașini ce preiau din ce în ce mai mult din funcțiunile naturale ale omului, va trebui poate să luăm în considerație mai atent primul mod de cuplare dintre memorie și dispozitivele efectorii [2]. Distribuirea inteligenței către dispozitivele executive va crea o legătură indestructibilă, între funcție și structură, ce în natură se manifestă cu toată plenitudinea,

\*) Institutul politehnic din București.



atît de puternic încît în lucrarea deja citată se spune, pe bună dreptate : „...a nu mai gîndi cu mîinile echivalează cu pierderea unei părți a gîndirii, ...”.

În tehnosferă, în știința calculatoarelor în particular, a fost asigurată o evoluție foarte spectaculoasă, obiectiv necesară poate, pe baza disocierii funcției de structură, printr-o poziție precis delimitată a memoriei în raport cu suportul material. A sosit momentul să revizuiți acest mod de abordare pentru că am evoluat rapid către un punct de unde nu mai putem merge mai departe fără să regîndim global modul de abordare a domeniului. Este valabil, se pare, și în domeniul limitat al științei calculatoarelor ceea ce A. Leroi-Gourhan spunea despre evoluția civilizației : „Ruperea legăturii dintre specie și memorie apare drept unica soluție ce poate duce la o evoluție rapidă și continuă”. Se pare că da, dar pînă unde ?

Vor reuși cercetările în domeniul calculatoarelor de generația a cincea să reazeze raportul dintre dispozitivele efectorii (structură), memorie și funcții astfel încît pe o bună perioadă de timp să mai poată fi marcați pași notabili în conceperea sistemelor inteligente, în conceperea sistemelor ce asigură și *exteriorizarea*, se pare, a ultimei funcții specific umane ce se poate preta la o astfel de acțiune ?

Cu riscul de a abuza în citarea unei singure lucrări, reținem următorul pasaj : „Pe un plan mai pozitiv, se constată că omul se îndreaptă treptat către exteriorizarea unor facultăți tot mai elevate, ca să poată profita la maximum de libertate, evitînd totodată riscul supraspecializării organelor sale.” Dacă omul evită supraspecializarea în evoluția sa, cum trebuie să devină calculatoarele, la nivelul lor, astfel încît în tendința spre adevare să-și păstreze flexibilitatea ? Cum se poate stabili echilibrul dintre *adevarea* și *flexibilitatea* funcțională ? Se pare că tot printr-un dozaj judicios al relației dintre structura efectorie și informația ce asigură adaptarea funcțională. În acest sens cred că trebuie să privim procesul de compatibilizare dintre funcție și structură în evoluția arhitecturii sistemelor de calcul. Modul în care se prefigurează gîndirea în domeniul calculatoarelor din noua generație credem că vine să sprijine acest punct de vedere.

## 2. Disjuncția funcție-structură

Într-un proces firesc de creștere, structurare, se pornește de la utilizarea unor configurații fizice ce — căpătînd virtuți funcționale — sînt utilizate la conceperea unor obiecte tehnice. Prima formă de concretizare a unui element funcțional este de tip structural. Cum evoluează, în tehnosferă, structura pe calea cuplării cu procesele informaționale în vederea augmentării virtuților funcționale ? La această întrebare încercăm un răspuns în această secțiune.

### 2.1. Automate cu spațiul stărilor structurat ( $AS^3$ )

Automatul elementar (A) este o structură de prelucrare a căre funcție este dată de o configurație fizică *regulată*, standard, registrul (R) și o configurație *adaptabilă* funcției, circuitul logic combinațional (CLO) prin care se închide bucla. Formal este definit prin :

$$A = (X, Y, Q, \lambda, \delta),$$



unde  $X$  este mulțimea variabilelor de intrare,  $Y$  cea a variabilelor de ieșire,  $Q$  este mulțimea stărilor, iar cele două funcții definesc tranziția stării ( $\lambda: X \times Q \rightarrow Q$ ) și tranziția ieșirii ( $\delta: X \times Q \rightarrow Y$ ). Funcțiile sînt efectuate de către un CLC cuplat în buclă cu un R. Sistemul astfel obținut este un sistem de ordinul doi (SO2), conform [21, 22]. În procesul de optimizare a unui A este utilizat, de regulă, conceptul de semiautomat (SA) deoarece optimizarea în spațiul ieșirilor nu este permisă într-un context dat. Prin definiție

$$SA = (X, Q, \lambda),$$

cu semnificațiile din definiția anterioară.

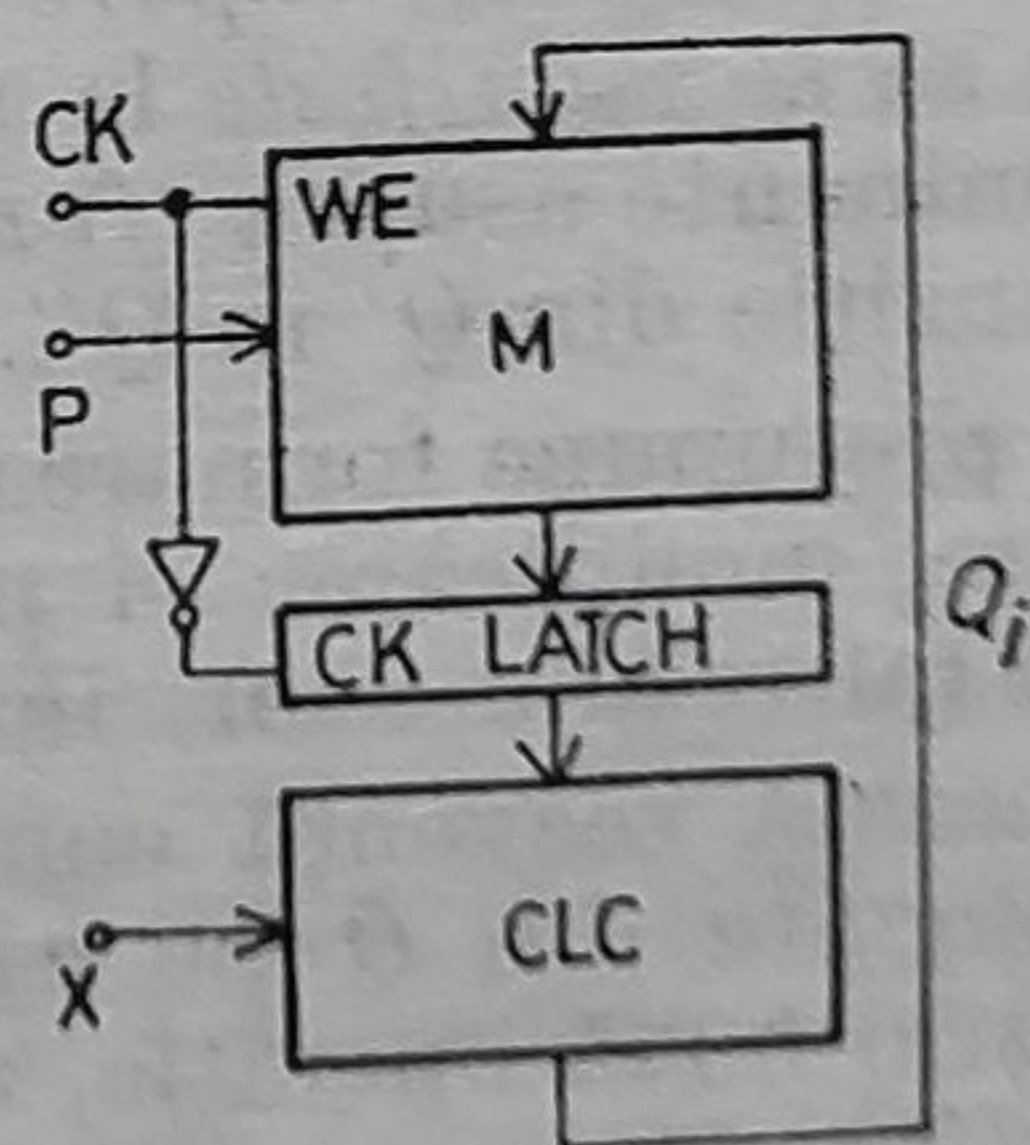
Din considerente de optimizare a funcției de tranziție  $\lambda$ , a fost introdus conceptul de automat cu spațiul stărilor structurat (AS<sup>3</sup>) [21, 22], ce presupune un semiautomat de tipul

$$SAS^3 = (X \times P, Q_0 \times Q_1 \times \dots \times Q_m, \lambda),$$

unde  $\lambda: X \times Q_i \rightarrow Q_i$  pentru  $P_i \in P$ . Mulțimea  $P$  indică, prin valoarea curentă, elementul produsului cartezian  $Q = Q_0 \times Q_1 \times \dots \times Q_m$  care este afectat în ciclul dat. Tranziția automatului din starea  $Q'$  în  $Q''$  se va realiza, de regulă, printr-o *secvență*. În figura 1 este prezentată schema de principiu a unui SAS<sup>3</sup> în care M este o memorie RAM adresată prin codul P iar LATCH-ul permite formarea unui registru împreună cu latch-ul din M selectat. Conținutul memoriei M formează o *structură informațională* (SI).

Între elementele SI există numai *legături sintactice*, de relație. De asemenea, valorile particulare ale elementelor din SI (elementele produsului cartezian  $Q$ ) nu au nici o semnificație pentru funcția sistemului.

Fig. 1. — Semiautomat cu spațiul stărilor structurat (SAS<sup>3</sup>).



Singura rațiune a apariției SI este *eficiența sistemului*, dobîndită prin minimizarea CLC în condiții în care timpul de prelucrare (tranziția  $Q' \rightarrow Q''$ ) crește în limite acceptabile.

## 2.2. Procesorul

Obținerea unui element de procesare presupune conectarea unui AS<sup>3</sup> cu un A de *control* în vederea secvențării procesului de comutare din  $Q'$  în  $Q''$ . O astfel de configurație este prezentată în figura 2, unde intrările lui AS<sup>3</sup> sînt  $X = X_1 \times X_2$ , ieșirile sale sînt  $Y = Y_1 \times Y_2$ , iar ieșirile lui A sînt  $P \times X_2$  și intrările  $Y_2 \times Q$ , mulțimea  $Q$  fiind cea a ordi-



nelor transmise sistemului de procesare. Se obține astfel un element de procesare cu două intrări și o ieșire ca SO3, ce ar putea sta la baza unei noi serii de structurări așa cum poarta logică cu două intrări și o ieșire a inițiat un prim proces de structurare ca element al SO0. Ce semnificație au elementele mulțimii  $Y_2$ ? De răspunsul la această întrebare

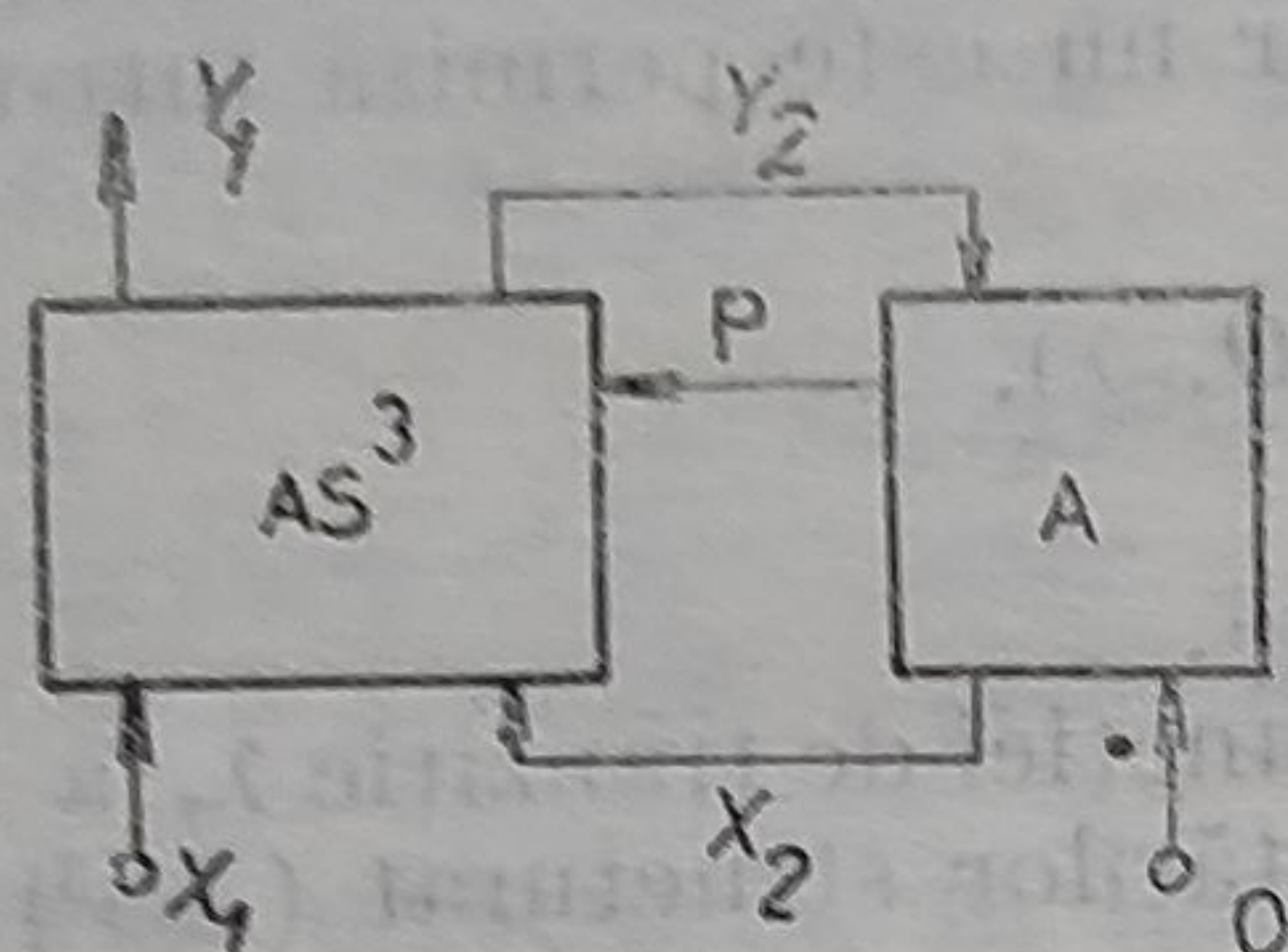


Fig. 2. — Structura generală a unui procesor.

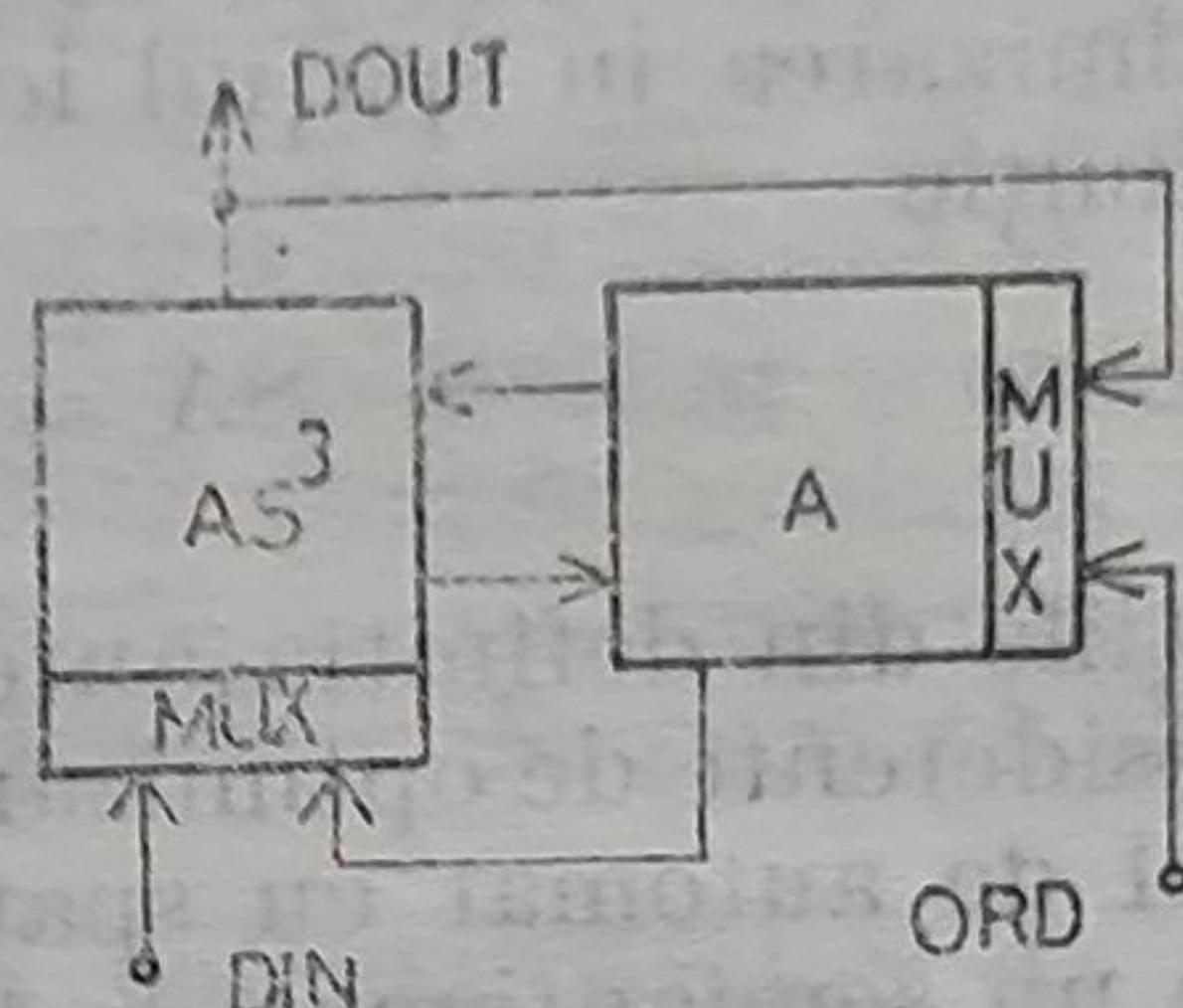


Fig. 3. — Suportul fizic pentru închiderea buclei prin structura informațională.

depinde extensia pe care o poate lua clasa de funcții acoperită de elementul de procesare. Dacă  $Y_2$  conține configurații binare ce caracterizează procesele din  $AS^3$  în baza cărora secvențarea comandată din A evoluează pe diverse căi, atunci sîntem într-un caz simplu al unui procesor în care rolul SI în determinarea funcției nu este important. Dar, dacă  $Y_2$ , printr-un cîmp al său, conține elemente din SI utilizabile în declanșarea unor operații univoc precizate în A, putînd înlocui ordinele primite pe calea O, atunci procesorul trece parțial sub controlul SI.

În cel de-al doilea caz se petrece un eveniment notabil, prin aceea că unele elemente ale SI capătă o semnificație precisă pentru funcționalitatea ansamblului. Dacă SI apare la nivelul sistemelor de ordinul doi (SO2), ea începe să se manifeste la nivelul celor de ordinul trei (SO3) cu consecințe determinante asupra funcției sistemului, asupra secvențării procesului de tranziție din  $Q'$  în  $Q''$ .

Procesarea presupune formarea unei SI prin achiziția unor șiruri de date de pe intrare, modificarea ei printr-o secvență comandată de A și emiterea unui șir (al rezultatelor) în exterior.

Dacă în această secvență complexă A poate modifica, chiar prin setare, unele elemente din  $Q$  pe care în continuare le folosește pentru a-și comanda sieși o nouă operație, atunci SI va fi direct implicată prin semnificațiile explicite ale unora din elementele sale în funcția de prelucrare a procesorului.

Concretizînd unele detalii din figura 2, în figura 3 se explicitează modul în care SI din  $AS^3$ , formată și de A, poate contribui la concretizarea secvențelor generate de A. Multiplexorul (MUX) din  $AS^3$  permite setarea unor elemente ale SI, iar cel din A generarea unor ordine către A utilizînd aceleași elemente din produsul cartezian ce definește starea complexă a  $AS^3$ . Se închide astfel o buclă în procesor prin SI a sistemului, „găzduită” în  $AS^3$ .

Apariția acestei noi bucle nu a fost condiționată de o nouă buclă structurală. Bucla între  $AS^3$  și A există strict condiționată de controlul  $AS^3$  prin A și nu pentru a facilita bucla prin SI. Această nouă conexiune necondiționată strict structural va conferi însă SI o nouă calitate.



### 2.3. Informația

Puține sînt conceptele care să aibă o utilizare atît de extinsă, în poate prea multe domenii, în condițiile în care sfera sa de semnificații este atît de puțin precizată, ca *informația*.

În lucrările [10] și [13] profesorul Mihai Drăgănescu, pornind și de la [11, 12], face o primă tentativă de definire generală a conceptului de informație spunînd că „...sub o formă destul de generală *informația* în raport cu mintea umană poate fi privită ca o structură căreia i se asociază un înțeles (meaning)”. Formal acest lucru este marcat prin

$$\mathcal{I} = \langle S, \mathcal{J} \rangle$$

unde:  $\mathcal{I}$  reprezintă informația,  $S$  — o structură sintactică concretă,  $\mathcal{J}$  este înțelesul ce este dezvoltat sub forma

$$I = \langle \mathcal{S}, \mathcal{R}, \mathcal{S} \rangle,$$

expresie în care:  $\mathcal{S}$  reprezintă semnificația de context,  $\mathcal{R}$  — semnificația de referință, iar  $\mathcal{S}$  are un caracter fenomenologic și reprezintă sensul asociat proceselor mentale.

Fără a intra în detalii, vom considera că forma cea mai simplă sub care se poate prezenta informația este

$$\mathcal{I}_0 = \langle S, \mathcal{R} \rangle,$$

structurii sintactice  $S$  fiindu-i obligatoriu asociat cel mai simplu înțeles: semnificația de referință.

O formă de tipul

$$\mathcal{I}' = \langle S \rangle$$

este considerată de M. Drăgănescu că „...reprezintă o „informație” pe care o vom numi *informație sintactică*”. Utilizarea ghilimelelor pentru termenul de informație din citatul anterior arată faptul că informația sintactică nu este încă informație.

În accepțiunea noastră, ținînd cont de cele expuse în paragraful anterior, *informația sintactică* poate corespunde structurii informaționale despre care am spus că posedă elemente ce nu au asociată nîci o semnificație pentru funcția structurii în care este inclusă.

Dar, în cazul în care elemente ale SI sînt prinse într-un proces de semnificare la nivelul unui procesor (SO3), putem spune că aceasta degerează în *informație*. Într-un procesor elementar SI dobîndește parțial capacitatea de a se manifesta semnificînd.

În consens cu M. Drăgănescu dar pornind de jos în sus considerăm că manifestarea incipientă a informației apare în cazul în care o structură (structura informațională) cu rudimentare legături sintactice dobîndește, parțial sau total prin fiecare element, în mod univoc o semnificație pentru evoluția unei mașini digitale. SI în manifestare este informație. Informația apare printr-un proces de semnificare ce se desfășoară în interacția dintre două automate.



Informația declanșează mecanisme interne, într-un sistem digital ce are cel puțin ordinul trei, într-o manieră ce nu implică structura fizică, care oferă numai un cadru de acțiune [21, 22]. Din această perspectivă apare rolul funcțional al proceselor informaționale. Informația poate fi privită ca un ingredient ce poate modifica funcționalitatea unui sistem în absența unor modificări în structura fizică.

Ca o consecință a faptului că stările unui automat pot fi precizate numai cu aproximația unui izomorfism, procesul de semnificare are un caracter arbitrar.

Filiația : stare internă (în A) — spațiu al stărilor structurat (în  $AS^3$ ) — informație (în procesor), poate da o primă imagine referitor la natura „obiectelor” ce constituie suportul structurii sintactice S din dubletul  $\langle S, J \rangle$ . Aceste „obiecte” sînt un fel de *mărimi de stare cu implicații funcționale* esențiale în sistem. Nu pot fi simboluri, semne, deoarece sînt definite cu aproximația unui izomorfism. Dacă ar fi semne, teoria informației ar fi o semiotică. Nu numai procesul de semnificare este arbitrar dar chiar procesul de *specificare* al S (în accepțiunea din [21] a SI) este arbitrar, rezultînd, în sistemele tehnice, dar poate și în altele, posibilități de manipulare folosibile pentru optimizarea structurală.

Caracterul arbitrar al specificării și al alocării semnificației de referință face ca informația să fie purtată de entități ce nu sînt strict simbolice.

Forma de manifestare cea mai evidentă a informației este oferită de *dinamica funcțională* ce se poate evidenția în absența unei dinamici structurale. În acest sens procesorul a evoluat degenerînd în structura clasică de calculator. În [21] este prezentată o cale posibilă pe care acest proces a avut loc. Rezultatul este o structură tripartită : procesor (P), canal de comunicație (C) și memorie (M) într-o configurație în care o mașină ce poate tinde către a fi *universală*, procesorul, se poate adapta unei *diversități funcționale* extraordinare datorită informației din memorie (fig. 4).

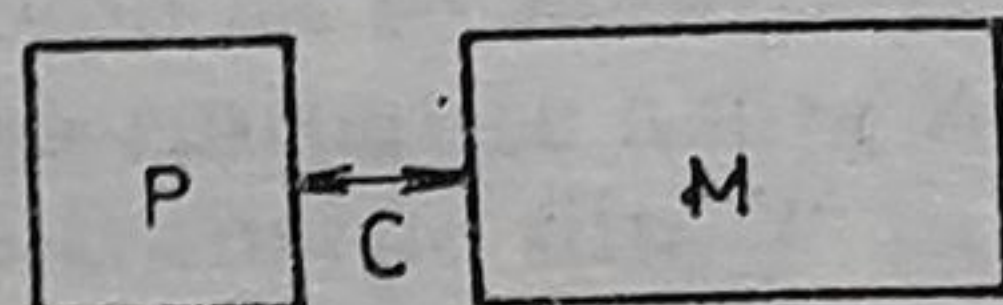


Fig. 4. — Modelul PCM ca mașină universală.

#### 2.4. Procesor-comunicație-memorie

Există un aspect fizic al tripletului P, C, M și unul conceptual. Fizic este vorba de un procesor, ca mașină universală avînd un set de funcțiuni prin care se poate adapta mai simplu sau mai dificil pentru a efectua orice operație, de o memorie de tip RAM *nestructurată* și de un canal cu o capacitate limitată de transfer. Conceptual, procesarea presupune două aspecte fundamentale : procesul de evaluare propriu-zis ; procesul de structurare a dispozitivului de memorare RAM în vederea dotării acestuia cu o funcție de memorie oarecare.

Accesul la memorie, din punct de vedere conceptual, presupune un proces desfășurat în ansamblul mașinii fizice formată din P, C și M. Canalul de comunicație nu este solicitat numai pentru *accesul* la o structură de date dar și pentru *structurarea* acesteia din urmă ; el reprezintă o gîtuire în principal datorită acestei *duble solicitări*.

Important nu este deci să înlăturăm limitările fizice introduse de conexiunea dintre P și M ci să eludăm efectul său în gîndirea sistemelor



de calcul. Conceptual, efectul este mai supărător decât sub aspect structural. Nu ne putem permite înlăturarea completă a funcției de comunicare dar putem să-i modificăm rolul în funcționalitatea sistemelor de calcul.

„John von Neumann bottleneck” este un termen [5] ce vizează în primul rând aspectele conceptuale ale conexiunii în mașina clasică. Un aspect, cel al structurării memoriei, a fost arătat, dar mai pot fi puse în evidență și altele.

## 2.5. Izolarea funcției de structură

Delimitarea netă între elementul efector (P) și mediul de stocare a informației a permis „o evoluție rapidă și continuă”. În condițiile în care P tinde către o structură cât mai *uniformă și compactă*, prin intermediul programelor stocate în memorie, funcțiunile se *diversifică* foarte mult. La limită orice problemă formal definibilă poate fi realizată. Programele stocate în memorie *mijlocesc* atât de bine adaptarea dintre funcțiile simple ale P și aplicațiile concrete încât *structura fizică se vede perfect izolată de funcția pe care o execută*.

Această *disjuncție netă între funcție și structură* este la originea limitărilor esențiale ale calculatorului văzut atât fizic cât și conceptual ca un triplet procesor-comunicare-memorie.

Funcția este strict mijlocită de programe, de structură informațională ce se manifestă ca informație. Mijlocirea se realizează însă într-un context în care informația și mașina efectorie sînt, de asemenea, distincte și suportate fizic de structuri net diferite.

La limită funcția este determinată strict de programe în virtutea tendinței de a identifica distincția structură-funcție cu distincția structură efectorie (procesor)-memorie.

SI apare, cel puțin în sistemele tehnice, pentru a optimiza structura efectorie (bucla combinațională a unui automat) și sfîrșește prin a prelua controlul asupra funcțiilor sistemelor digitale.

Într-o lucrare [22], numeam acest lucru : proces de structurare neechilibrată. Conceperea unor calculatoare cu memorii din ce în ce mai mari și cu procesoare din ce în ce mai compacte, eventual mai rapide, nu poate fi luată în considerație pentru viitor. Realizarea unor mutații măcar la nivelul triadei P—C—M este iminentă și ultimele experiențe o dovedesc. Pentru a le putea pune în evidență trebuie să căutăm corelații mai profunde între *procesul de structurare fizică* și cel de alocare a unor *clase de funcții* diverselor nivele structurale. Din acest motiv secțiunea următoare va urmări legătura între limbaje și complexitatea structurală.

## 3. Limbaje și mecanisme de structurare

La sfîrșitul deceniului șase și începutul următorului, Noam Chomsky într-o serie de lucrări, între care și [6, 7], pune bazele formale ale limbajelor generative propunînd o clasificare a lor în funcție de restricțiile acceptate de gramaticile asociate. Limbajele din cadrul fiecărui tip au puse în corespondență o anumită clasă de mașini ce permit recunoașterea și generarea șirurilor ce le aparțin [1, 8]. În această secțiune încercăm



să arătăm relația ce există între limbaje, mașinile asociate și mecanismul de structurare a mașinilor digitale [21 — 23], pentru a putea găsi o corelație utilă între structură și funcția asociată.

### 3.1. Limbajele regulate și sistemele de ordinul doi

Începem prin a aminti că limbajele regulate, de tipul trei în clasificarea lui N. Chomsky, sînt caracterizate prin faptul că regulile de generare sînt de forma

$$A \rightarrow x B,$$

unde  $A$  și  $B$  sînt neterminale, iar  $x$  este un șir format din terminale.

Altfel spus, dacă, o producție  $p \rightarrow q$ , unde  $p$  și  $q$  pot fi șiruri conținînd simboluri terminale și neterminale, admite următoarele restricții:

—  $|p| \leq |q|$ , dimensiunea șirului generat nu poate fi mai mică decît a celui generator în orice etapă a procesului de generare;

—  $|p| = 1$ ;

—  $q = \alpha A$ , unde  $\alpha$  este un șir format din simboluri terminale; atunci, regula de generare aparține unei gramatici regulate.

Pentru a genera un șir într-un limbaj de tip 3 ( $\mathcal{E}_3$ ) *producțiile se aplică într-o ordine reflectată de ordinea simbolurilor din șirul rezultat*. Această ordonare permite recunoașterea și generarea printr-un proces similar ordonat, *nefiind necesară o memorie asociată mașinii*.

Este utilizat din acest motiv numai un automat finit,  $A$ , (sistem de ordinul doi — SO2) pentru recunoașterea și generarea secvențelor regulate. Acest automat nu are asociată nici o memorie externă care să sprijine funcționarea sa. Este suficientă funcția de memorie asigurată de traiectoria prin spațiul stărilor automatului. Funcția de memorie, în măsura rudimentară în care există, este *intrinsec asociată elementului efector*, automatul finit.

### 3.2. Limbajele independente de context și sistemele de ordinul trei

Limbajele de tipul 2, independente de context, posedă reguli de generare mai puțin restrictive, deci, vor presupune, pentru generare și recunoaștere, mașini mai complexe. Prin definiție, o producție într-o gramatică de tipul 2 este de forma

$$A \rightarrow \alpha$$

unde  $A$  este un neterminal iar  $\alpha$  un șir ce conține terminale și neterminale. O altă modalitate de definire presupune existența numai a două restricții în delimitarea regulilor de generare. Acestea sînt:

—  $|p| \leq |q|$ , șirul crește prin generare;

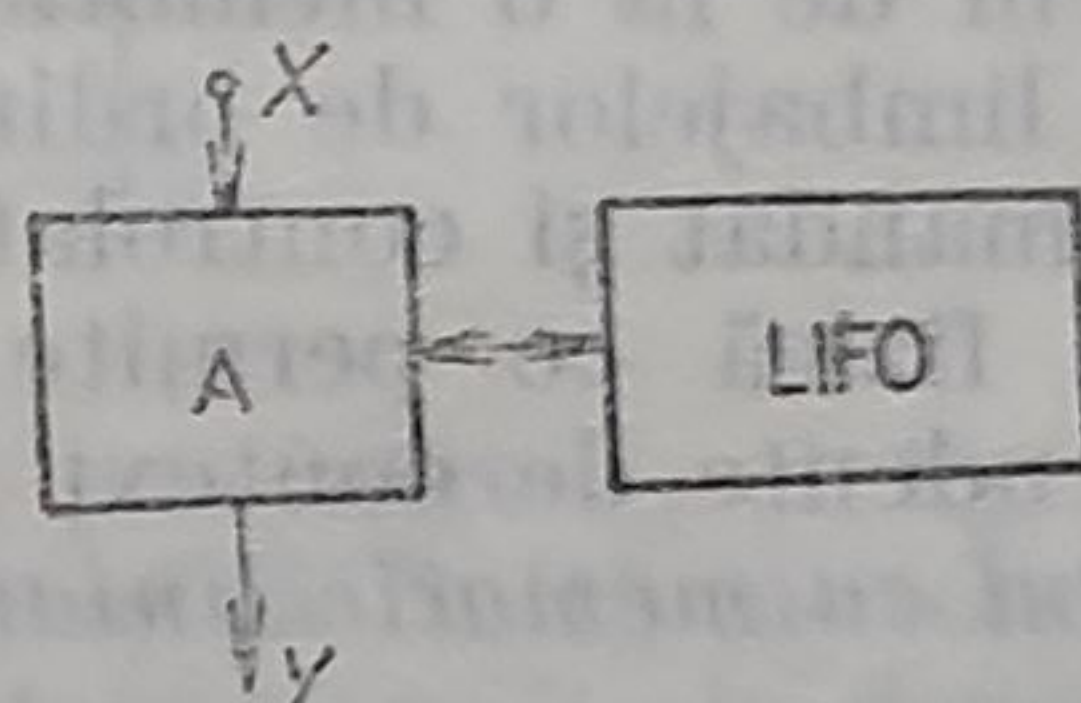
—  $|p| = 1$ , indicînd că un simbol,  $p$ , se înlocuiește complet necorelat cu simbolurile ce-l delimitează (generarea este independentă de context); evident,  $p$  este un neterminal.



Remarcăm faptul că elementele unui șir *nu apar în șirul final într-o ordine precizabilă*, aplicarea regulilor ce au determinat apariția fiecărui element presupunând un proces de „du-te-vino” printre elementele șirului parțial generat.

Dacă, în cazul expresiilor regulate pe măsură ce elementele șirului erau aplicate intrării automatului, acesta putea decide dacă șirul aparține sau nu limbajului, generat de o anumită gramatică, în cazul expresiilor dintr-un limbaj independent de context cel puțin porțiuni din șirul de recunoscut trebuie stocate înainte de a se putea decide apartenența sau nu la un limbaj, prin confruntarea cu alte subșiruri. Această primă obser-

Fig. 5. -- Automat cu stivă.



vație impune *adăugarea unei memorii automatului finit de recunoaștere*. Structura acestei memorii este foarte simplă, este poate cea mai simplă, anume cea de LIFO. Se demonstrează că un *automat cu stivă* este capabil să recunoască sau să genereze expresii aparținând limbajelor independente de context.

Un automat cu stivă este un *sistem de ordinul trei (SO3)* deoarece în bucla unui SO2, automatul, se conectează un SO1, memoria LIFO, conform reprezentării din figura 5. Mașina asociată limbajelor de tip 2 s-a complicat suficient de mult ca să devină un sistem de ordin imediat superior lui SO2. Remarcăm că apare pentru prima dată perechea element de prelucrare (A) și memorie (LIFO) în șirul mașinilor asociate ierarhiei limbajelor.

### 3.3. Limbaje dependente de context și sistemele de ordinul patru

Producțiile admise de o gramatică dependentă de context sînt de forma

$$\alpha \rightarrow \beta,$$

unde  $\alpha$  și  $\beta$  sînt șiruri nevide formate din terminale și neterminale ce îndeplinesc numai condiția ca

$$|\alpha| \leq |\beta|.$$

O altă formulare posibilă este aceea ce spune că producțiile sînt de forma

$$\alpha_1 x \alpha_2 \rightarrow \alpha_1 p \alpha_2,$$

unde  $x$  este un neterminal iar  $p$  un șir nevid de terminale și neterminale. Șirul  $p$  este generat de  $x$  în *contextul* oferit de șirurile  $\alpha_1$  și  $\alpha_2$ , și numai în acest context.

Dacă recunoașterea șirurilor generate de gramatici mai restrictive se poate face într-un proces ce se desfășoară și pe măsură ce șirul este



recepționat, în cazul șirurilor aparținând unui limbaj de tipul unu este necesară *memorarea, prealabilă procesului de recunoaștere*, a întregului șir. Efectul oricărei producții posibile trebuie analizat în contextul oferit, la limită, de întregul șir sau de subșiruri arbitrare specificate. De asemenea, generarea unui șir prin reguli ale unei gramatici dependente de context, trebuie să fie *complet încheiată* într-o memorie internă *înainte ca șirul să fie emis* în exterior ca rezultat al procesului de generare.

Accesul la memorie nu mai poate avea caracterul ordonat (de tip LIFO), regulile de acces trebuie să permită un acces *aleator* pentru a se putea pune în evidență *orice context* impus de regulile de producție. Va fi nevoie să trecem de la o memorie simplă, cu un mecanism *implicit* de acces în cazul limbajelor de ordinul doi, la o memorie la care accesul este *explicit* comandat și controlat.

Structura fizică ce permite recunoașterea și generarea șirurilor limbajelor dependente de context este, precum se demonstrează în literatură, *automatul cu memorie liniar mărginită*. În figura 6 este reprezen-

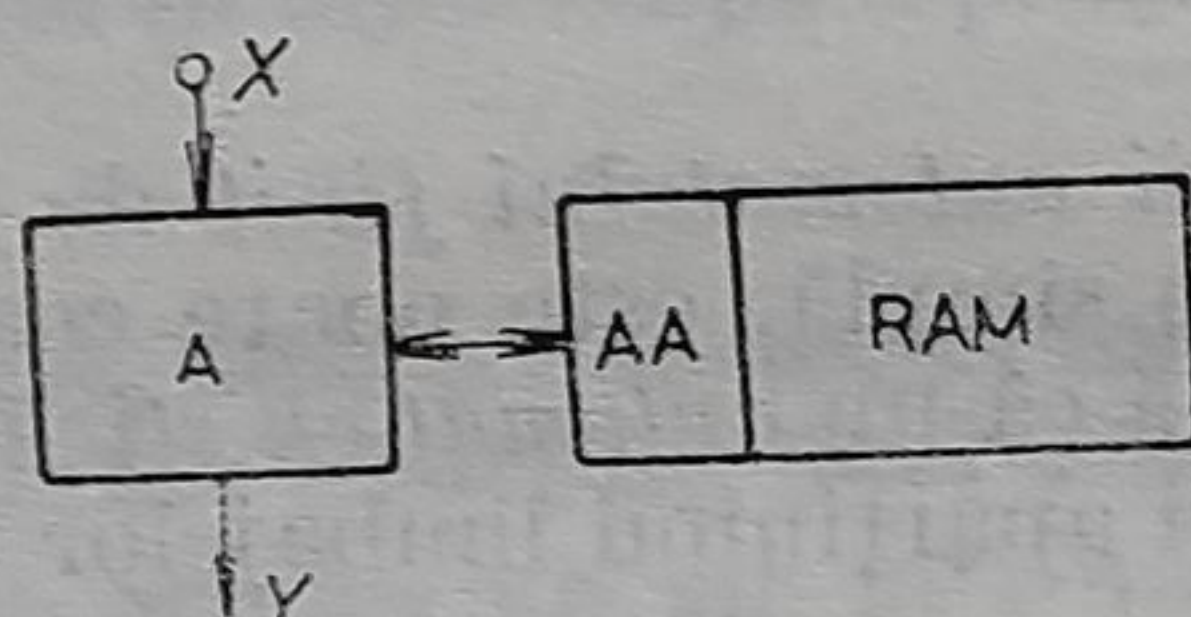


Fig. 6. — Automat cu memorie liniar mărginită.

tată o astfel de mașină ce conține, atașat automatului A, memoria liniar mărginită formată dintr-un automat de adresare AA și o memorie RAM. Cuplul AA + RAM este un SO3 deci întregul ansamblu este un SO4 ce are ca reprezentant tipic conceptul de *calculator*, așa cum SO3 avea ca reprezentant tipic *procesorul*.

Asociem deci clasa  $\mathcal{E}_1$  cu SO4.

Trebuie remarcat faptul că declanșarea sau terminarea proceselor de recunoaștere, respectiv, de generare, presupunând *șirul* de recunoscut sau generat *în memoria internă*, sînt create *premisele* declanșării unor *procese paralele* pentru ambele tipuri de funcțiuni. Declanșarea acestora este *împiedicată* însă de *natura regulilor* de producție în clasa limbajelor de tipul unu. Fiecare regulă presupunând un anumit context, modificarea acestuia în paralel prin aplicarea altei reguli de producție creează indecizie în stabilirea contextului în care se aplică prima regulă. *Înlănțuirea* aplicării producțiilor este o cerință ce nu poate fi eludată. Deci, *chiar dacă structura hardware creează premisele paralelismului funcțiunea poate să se opună actualizării lui*.

Mașina asociată  $\mathcal{E}_1$  este, la modul general, formată tot dintr-un element efector, A, și o memorie, AA + RAM, dar efectul conceptual al conexiunii dintre ele este minimizat deoarece de structurarea memoriei se ocupă AA, A fiind degrevat de această sarcină ce s-ar adăuga celei de control al proceselor de recunoaștere sau generare.

### 3.4. Ce conține spațiul pînă la mașina Turing?

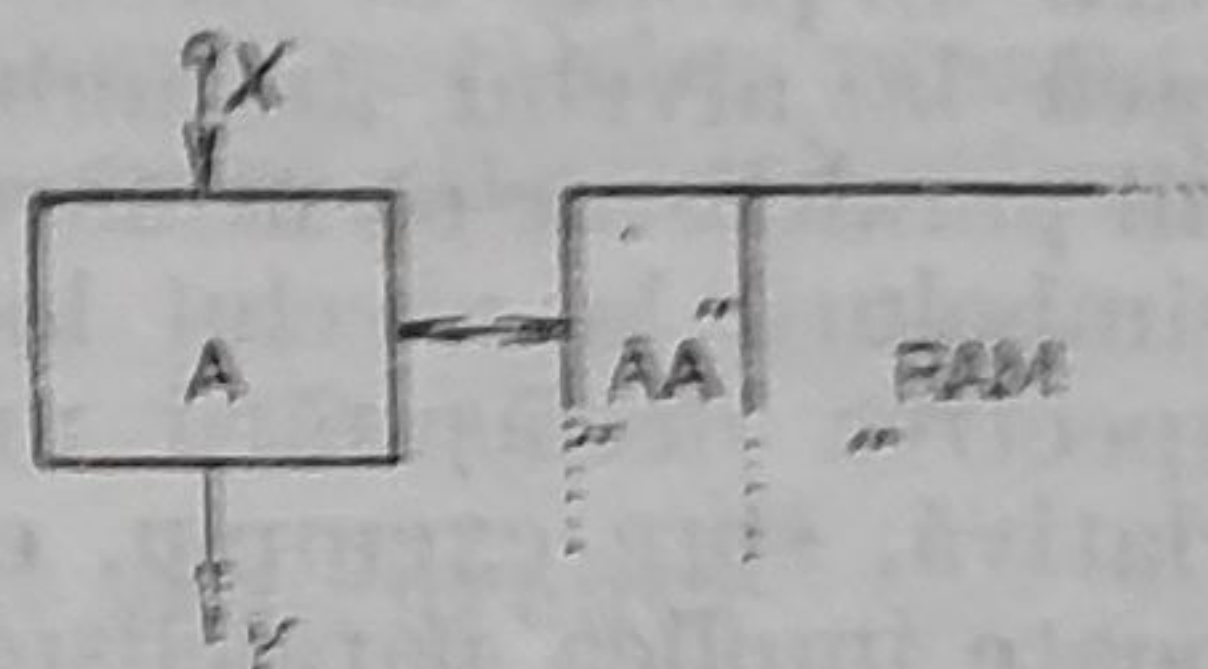
În absența *oricăror* restricții impuse regulilor de generare se obțin șiruri ce nu pot fi recunoscute decît de mașina Turing. Structura unei mașini Turing considerăm că nu poate fi corect specificată, de aceea pre-



ferăm să o reprezentăm ca în figura 7 deoarece este greu să ne imaginăm, credem că este imposibil, un automat ce adresează, gestionează, o memorie infinită, iar o memorie infinită nu sîntem siguri că mai poate fi asimilată cu un RAM infinit.

N. Chomsky face însă un singur pas între  $\mathcal{L}_1$  și  $\mathcal{L}_0$  pe care o asociază mașinii Turing. Este adevărat că Marvin Minsky în 1967, într-o carte

Fig. 7. — Mașină Turing.



altfel excepțională [8], făcea un singur pas între  $\mathcal{L}_1$  și  $\mathcal{L}_0$ , între limbajele regulate și cele fără restricții : între o mașină finită (automatul elementar) și una infinită (mașina Turing). Ne putem permite în deceniul nouă întrebarea : ce se află între  $\mathcal{L}_1$  și  $\mathcal{L}_0$ ?, între automatele cu memorie liniar mărginită și ipotetica, prin aceasta, fascinantă, mașină Turing?

În virtutea paralelismului evidențiat între limbaje, mașini și niveluri de structurare, cred că putem fixa următoarele corespondențe :

- |   |        |
|---|--------|
| $\mathcal{L}_3$ — automate finite               | — S02, |
| $\mathcal{L}_2$ — automate cu stivă             | — S03, |
| $\mathcal{L}_1$ — automate cu memorie mărginită | — S04, |

prin care încercăm încă o dată validarea mecanismului de structurare prin bucle, expus prima oară în [23].

Pare legitimă întrebarea : ce limbaje corespund la SON, unde N este diferit de 2, 3 și 4?

Dacă pentru  $N < 2$  ne putem aștepta la limbaje mult *prea restrictive* condiționate pentru a fi utile, pentru  $N > 4$  se deschide un spațiu în care s-ar putea defini limbaje din ce în ce *mai puțin restrictive* și prin aceasta din ce în ce mai adecvate descrierii unor realități mai nuanțate, mai complexe, într-o manieră mai apropiată de modul *natural de specificare*.

Din punctul de vedere al memoriilor atașate automatului se pot pune în evidență corespondențele :

- |   |
|---|
| $\mathcal{L}_3$ — memorie nulă,             |
| $\mathcal{L}_2$ — memorie LIFO,             |
| $\mathcal{L}_1$ — memorie liniar mărginită, |

remarcînd faptul că dacă A nu evoluează în trecerea de la  $\mathcal{L}_3$  la  $\mathcal{L}_0$ , memoria devine din ce în ce mai complexă sub aspect funcțional. În cadrul perechii (A + memorie) structurile debutează prin a fi fără memorie dar vor putea evolua către structuri în care rolul memoriei să fie dominant, memoria putînd deveni o funcție din ce în ce mai complexă.

Se pot imagina limbaje mai puțin restrictive decît  $\mathcal{L}_1$ , care să nu fie însă de tip  $\mathcal{L}_0$ , astfel încît mașina ce le-ar recunoaște sau genera să necesite funcțiuni mai complexe de memorare, cum ar fi : memorii arborescente, memorii asociative, poate chiar baze de date relaționale. Raportat la o astfel de perspectivă rolul elementului efector, automatul A, tinde să fie nesemnificativ în ansamblu, mașina tinzînd către o memorie cu funcționalitate complexă.



### 3.5. Procesul de migrare funcțională

Dacă într-o structură clasică aspectele funcționale legate de structură se concentrau în elementul de procesare, ce în cel mai simplu caz poate fi un automat finit, în perspectiva definirii unor limbaje din ce în ce mai nerestrictive, mai „expresive”, funcțiile tind să migreze către zona de memorare într-un proces de structurare echilibrată.

Chiar dacă la nivelul ansamblului *elemente de procesare-memorie* nu se pot defini paralelisme ca urmare a restricțiilor impuse de procesarea șirurilor de simboluri, la nivelul local, al funcțiilor de memorare este deschisă perspectiva desfășurării unor procese paralele. Chiar dacă o memorie asociativă, spre exemplu, este utilizată într-un proces de evaluare ce nu poate implica paralelismul, la nivelul funcției de memorare asociativă, deci al structurii ce o implementează, se vor putea declanșa nestingherite procese paralele.

Procesul de migrare funcțională creează premise ale paralelismului chiar în cazul unor limbaje unde acesta la nivelul evaluării nu poate fi presupus.

Variantele de mașini prezentate în [19] pot constitui un exemplu al evoluției către situația în care funcția de procesare degenerază, măcar parțial, în funcție de memorare într-o configurație ce păstrează aparența unei mașini clasice prin tripletul P-C-M, chiar dacă termenul median pierde o mare parte din rolul distructiv pe care-l are. În același sens poate fi citată și lucrarea [15] unde este prezentată arhitectura unei baze de date la nivelul căreia paralelismul este foarte avansat, dar care poate fi asociată unei arhitecturi mai complexe în care să apară restricții asupra proceselor paralele. Utilizarea structurilor aparent convenționale, dar la care efectul conceptual al C să fie puternic atenuat, pînă la limita la care caracterul convențional al mașinii să dispară, este încă actuală [24].

O primă tendință, deci în îndepărtarea de la modelul de tip P-C-M este aceea a *migrării funcției către structură* într-un proces de anulare a disjunției evidențiate în secțiunea a doua a acestei lucrări. Funcții implementate prin programe tind să fie implementate pe *structuri specializate* chiar dacă în acestea se mai folosesc tehnici de microprogramare sau nano-programare. Clasa de funcțiuni care, astfel implementată, produce efectele cele mai spectaculoase este cea a funcțiilor de memorie. Nu trebuie însă să neglijăm, chiar dacă nu am adus argumente în acest sens, efectele produse de conceperea și utilizarea coprocesoarelor numerice sau a interfețelor inteligente.

Dacă în structurile clasice numai *suportul fizic specific al funcției de memorare* se detașa de elementul de procesare, sub forma unui RAM de mări dimensiuni în care se structurau datele și programele, într-o abordare cu caracter neconvențional se detașează de procesor un element care conține funcțiuni de memorare, în sensul unor *memorii structurale*, nu amorf precum un RAM. La limita acestui proces mașina în ansamblu poate deveni o memorie foarte complex structurată în care să se regăsească sub forma unor funcții de memorare, ansamblul funcțiilor unui sistem de calcul. Ne vom putea pune în situația, deocamdată ipotetică, întrevăzută de Edward A. Feigenbaum în [14]: „The Basic Idea of Expert Systems: Putting Knowledge to Work.” Funcția de reprezentare a unei realități, chiar puternic mărginite, va putea conține implicit și



funcția de operare pe și cu această reprezentare. Așa cum la o funcție simplă de memorare accesul la memorie presupune un mecanism implicit de prelucrare, de ce nu, la o memorie de mare complexitate accesul de la memorie către exterior să presupună un proces de prelucrare deosebit de semnificativ. Încipient acest lucru este valabil pentru circuitele de memorie asociativă.

Menționind că funcțiunile de memorare pot presupune, independent de procesul de evaluare, mecanisme paralele; prezentăm o primă modalitate prin care migrarea funcției către structură stimulează declanșarea de procese paralele.

#### 4. Paralelism și concurență

Apropierea și adecvarea funcțiilor la structură creează premisele conceperii unor componente din ce în ce mai diverse, fapt ce va implica dezvoltarea sistemelor digitale în rețele. Ideea de rețea implică aproape natural *paralelismul* dacă se impune o utilizare eficientă a resurselor.

Paralelismul este impus și de necesitatea sporirii vitezei de execuție în condițiile în care limitările tehnologice și cele impuse de algoritm au fost atinse. Vom încerca să rezolvăm mai repede cu multe procesoare ceea ce nu am putut rezolva suficient de rapid cu unul singur.

Simpla alocare a unor resurse în paralel nu asigură viteza net sporită datorită limitărilor impuse de *concurență*. Aceasta se poate manifesta la *resursele fizice* ale sistemului sau la sursele de *operanți*, deoarece nu se poate, în toate situațiile, *alimenta ritmic* fiecare nod al rețelei.

O posibilitate de a clasifica rețelele este aceea după conținutul nodurilor. În acestea pot fi prinse elemente complete P-C-M ce au o funcție de procesare sprijinită de o memorie locală și asigură, de asemenea, funcția de canal de comunicare în rețea. O altă posibilitate o oferă noduri în care domină una dintre funcțiile de procesare, memorare sau comunicare, fără ca celelalte să poată fi complet atrofiate. Tehnologiile VLSI stimulează expandarea sistemelor sub formă de rețele omogen structurate pe suprafețe cit mai mari. Interconectarea unor astfel de rețele generează de regulă sisteme cu o funcționalitate complexă și un timp de execuție foarte mic.

##### 4.1. Rețele cu noduri de tip P-C-M

Primele tipuri de rețele omogene au fost structurate pornindu-se de la structuri clasice, în noduri fiind conectate mașini de tip P-C-M ce posedau posibilități locale de calcul relativ puternice și posibilitatea de a opera cu blocuri de date suficient de mari. Astfel de rețele sînt gîndite și pentru limbajele mai vechi de programare, astfel că ele nu evită concurența în primul rînd la memoria comună a sistemului. O configurație simplă a unei astfel de rețele este prezentată în figura 8, cele  $n + 1$  elemente de calcul formate dintr-o memorie  $M_i$  și un procesor  $P_i$  sînt în comun conectate la  $P$ , ce facilitează accesul la  $M$ . Se pot deosebi două cazuri semnificative, notînd cu  $C(M)$ , capacitatea de stocare a memoriei  $M$ ;

(1)  $C(M_i) \ll C(M)$  pentru  $i = 0, 1, \dots, n$ ,

(2)  $C(M_i) \gg C(M)$  pentru  $i = 0, 1, \dots, n$ .



Paralelismul oferit de cele  $n + 1$  perechi procesor-memorie va fi limitat de concurența la resursa comună P-M, care poate funcționa și ca un controler de sistem. Distincția între (1) și (2) a fost discutată în [22].

Resurse de același tip în rețea și în controlerul de rețea, în cazul nostru memorii, fac să fie stimulată concurența.

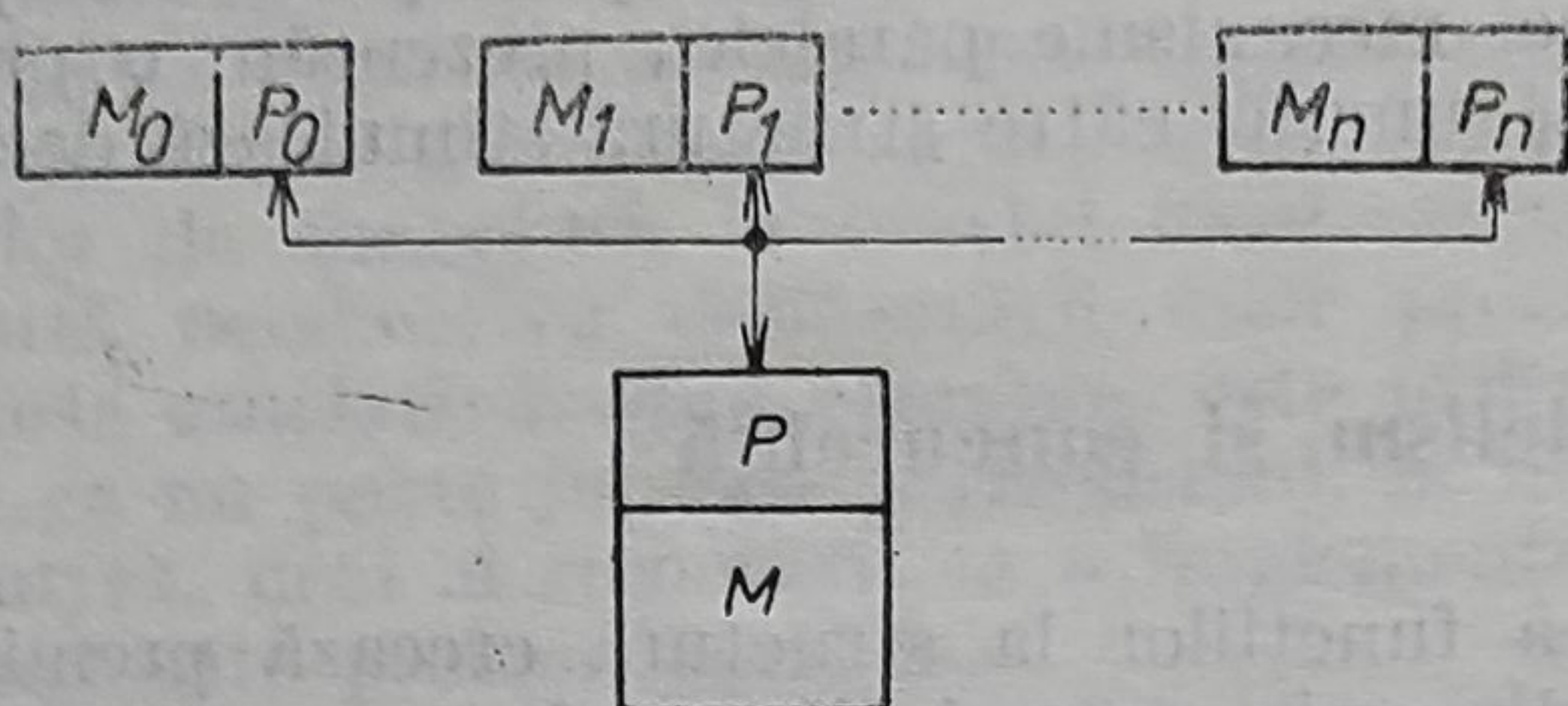


Fig.—8. Rețea omogenă cu elemente de tip PCM.

Funcțiile efectuate de  $P_i - M_i$  sînt condiționate și de fluxul de date către și de la  $M$ . Din acest motiv concurența într-o astfel de rețea poate deveni mare. În cazul în care  $C(M_i)$  ar fi foarte mică sau chiar nulă, funcția elementului din nod nu ar depinde de comunicarea cu controlerul  $P-M$ .

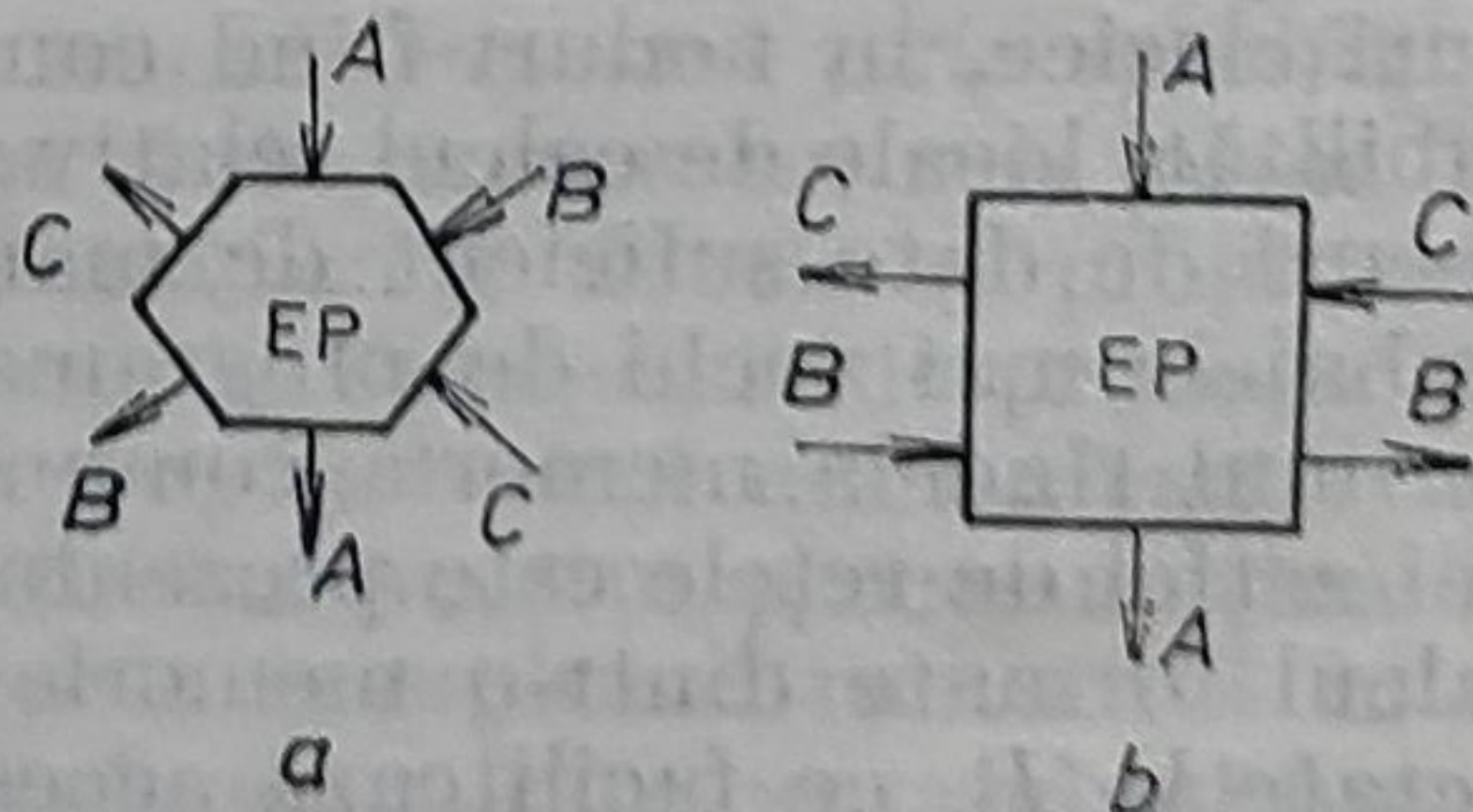
Există și funcțiuni pe care o astfel de rețea lucrează în condiții de concurență nulă. Spre exemplu, dacă ar implementa o memorie asociativă pentru mașina  $P-M$ , atunci la căutare s-ar declanșa un paralelism perfect. În cazul scrierii ar apărea o concurență la „operanzi” în sensul că numai o pereche  $P_i - M_i$  ar funcționa. Într-o memorie asociativă nu este însă critică funcția de scriere ci numai cea de citire. Putem exemplifica cu arhitectura descrisă în [15].

Reținem faptul că paralelismul presupune concurență nu în funcție de configurația unei rețele sau de conținutul nodurilor acesteia numai, ci în primul rînd de modul în care este folosită rețeaua, de funcția ei.

#### 4.2. Rețele cu elemente de procesare

Existența unor *resurse funcționale specifice* creează premisele unui paralelism neperturbat de concurența la resurse fizice. Un exemplu tipic

Fig. 9. — Element de procesare aritmetică pentru sisteme sistolice.



îl poate constitui o rețea omogenă cu elemente de procesare (EP) aritmetice de tipul celor din figura 9 [19]. Cu astfel de elemente se pot configura rețele unidimensionale sau bidimensionale care să permită înmulțiri de



vectors, de vectori cu matrice, de matrice cu matrice, de calcul al transformatei Fourier ș.a. Funcția asociată PE este

$$C \rightarrow C + A \times B,$$

$$B \rightarrow B,$$

$$A \rightarrow A$$

avînd o componentă de procesare, cea principală, și una ce asigură procesul de comunicare.

De regulă astfel de rețele omogene pot fi ușor gîndite pentru funcțiuni numerice, asigurîndu-se un paralelism foarte avansat în absența concurenței la resurse. Se pune însă problema alimentării cît mai *ritmice* cu operanți a elementelor din noduri și a obținerii unei rate constante de transfer prin rețea a operanzilor de tip matrice sau vector. Acest fapt implică găsirea unor algoritmi adaptați procesării într-o astfel de rețea. În literatură [25] se folosește termenul de *formă sistolică* a unui algoritm și de *sistem sistolic* pentru sistemul ce-l poate implementa.

O altă variantă de rețea o constituie varianta de tip pipe-line. Limitarea principală în acest caz apare datorită imposibilității declanșării proceselor ce presupun bucle. Recursivitatea într-o structură de acest tip nu poate fi declanșată decît cu restricții, de regulă, inacceptabile.

Odată cu impunerea ideii de programare funcțională [5], au mai fost utilizate două tipuri de structuri din această categorie: structurile ierarhizate [17] și structurile cu flux de date, [3, 4].

Un exemplu de mașină ierarhizată este reprezentat în figura 10. Ea a fost propusă de G. Mago [17] și conține două tipuri de elemente. Cele de tip T cu funcție preponderentă de comunicare asigură propagarea formelor funcționale către frunzele arborelui în vederea unei prelucrări mai complexe. Cele de tip L realizează prelucrări complexe, rezultatele fiind transmise prin celulele de tip T către tulpina arborelui. Funcția esențială a rețelei este de procesare, deoarece chiar celulele de tip T posedă unele funcții de prelucrare.

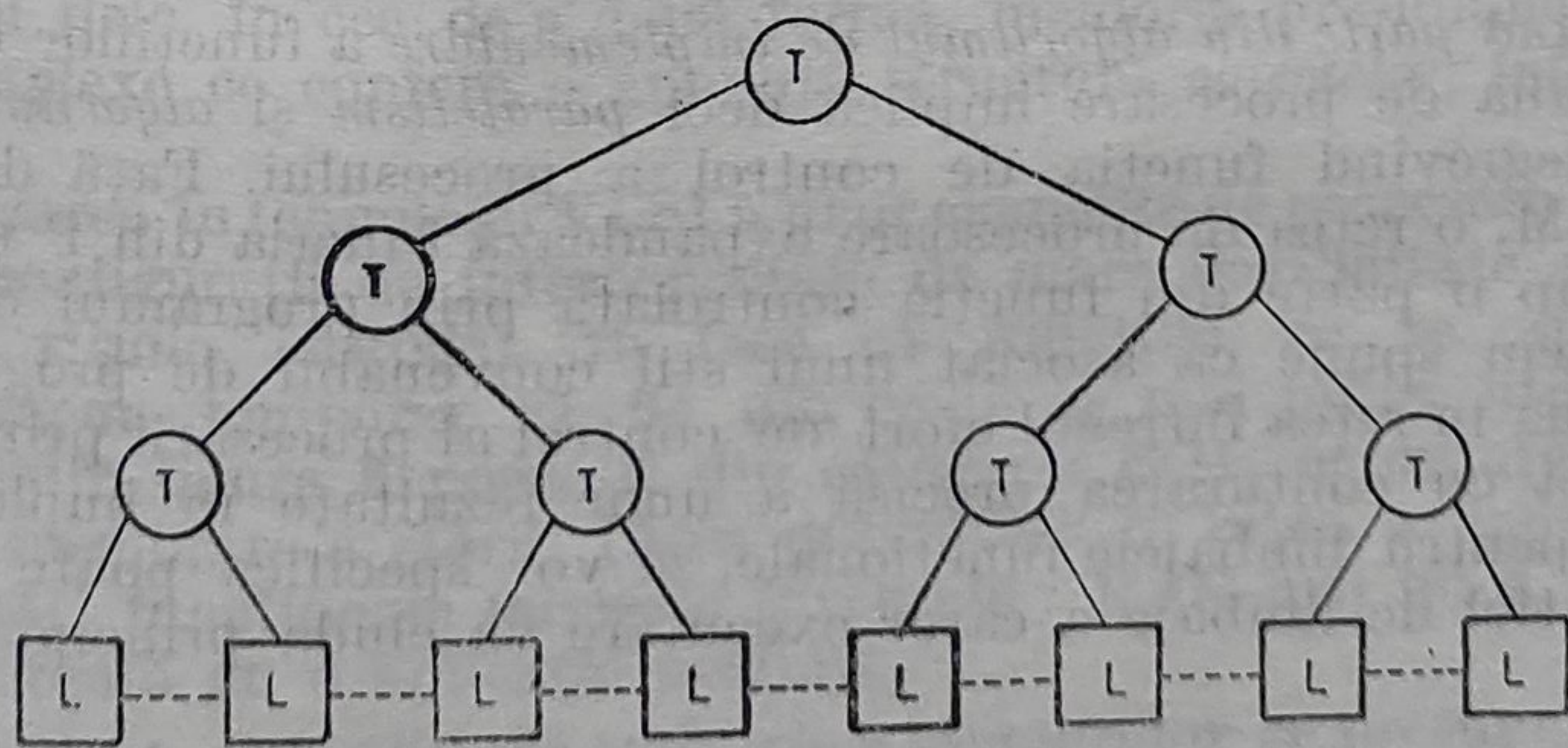


Fig. 10. — Exemplu de rețea ierarhizată (G. Mago).

Propunerea făcută de profesorul Arvind [3] separă mult mai net elementele de procesare (EP) de rețeaua de interconectare multiplă (vezi figura 11).

În structurile pipe-line și în cele omogene *fluxul datelor este determinat de modul de conectare* al elementelor rețelei. Spre deosebire de acestea,



în rețelele ierarhizate și cele cu flux de date *transferul este controlat de chiar informația transferată*. Problema funcționării în regim sistolic, în ultimele două tipuri de rețele, nu se poate pune deoarece regimul de

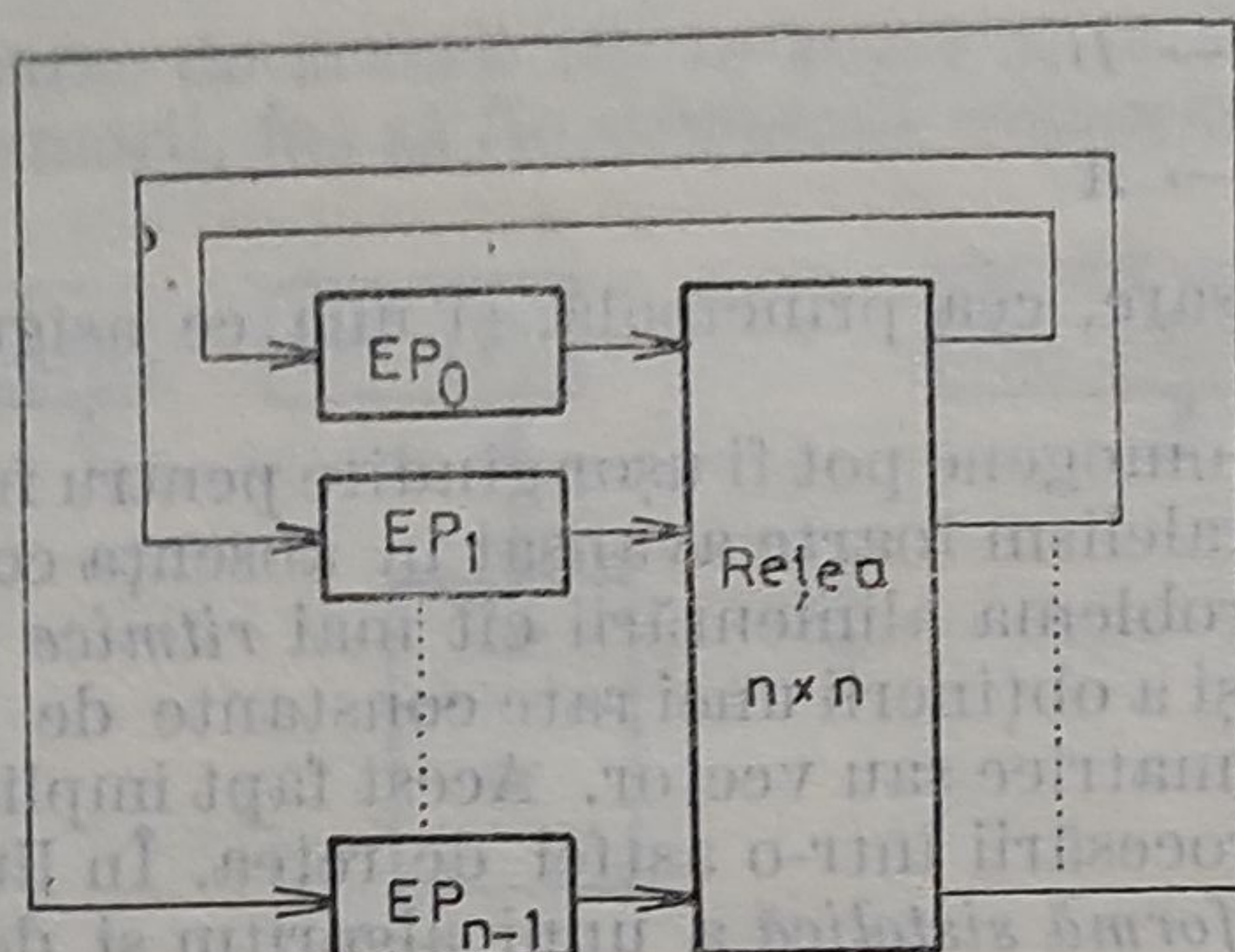


Fig. 11. — Structura propusă de prof. Arvind pentru limbaje funcționale.

solicitare al elementelor rețelei este dictat de configurația particulară a datelor. În primele două cazuri o parte din algoritm era cuprinsă în maniera de structurare a rețelei și în modul de injectare și extragere a datelor. În ultimele două exemple configurația rețelei nu influențează algoritmul, ce este impus exclusiv de forma funcțională evaluată.

Problema concurenței în structurile ierarhizate și în cele cu flux de date se rezolvă automat printr-o pulsație mai intensă sau mai puțin intensă a datelor în rețea. Dimensiunea formei funcționale nu trebuie corelată cu dimensiunea celor două rețele, consecința unei structuri reduse fiind numai lungirea timpului de evaluare.

Spre deosebire de acestea structurile pipe-line și rețelele omogene presupun o corelare a dimensiunilor rețelei și a modului de injectare și extracție, cu dimensiunea operanzilor.

Remarcăm faptul că structurile paralele de procesare nu au numai rolul de a mări viteza de prelucrare. Prin regulile lor de structurare, prin modul în care rezolvă problema comunicării între elementele de procesare, *preiau o bună parte din algoritmul de implementare* a funcțiilor executate.

Rețeaua de procesare implică deci *parallelism* și *algoritm* în egală măsură, degrevând funcția de control a procesului. Față de modelul clasic P-C-M, o rețea de procesoare expandează funcția din P și preia în același timp o parte din funcția controlată prin programul din M. La limită putem spune că asociat unui stil convenabil de programare se poate prelua în rețea întregul efort de control al procesării prin program.

Odată cu conturarea precisă a unor rezultate în implementarea mașinilor pentru limbajele funcționale, se vor specifica poate concret o serie de astfel de limbaje a căror executare va eluda principial modelul P-C-M.

#### 4.3. Rețele de interconectare

Principiul mașinilor bazate pe flux de date, expus pentru prima oară de Jack Dennis de la MIT [9], impune, odată în plus, și conceptul de rețea de interconectare, ca o componentă ce poate deveni foarte importantă într-un sistem VLSI. În figura 11 rețeaua este o componentă esențială,



ce asigură principal posibilitatea unui paralelism foarte avansat neimpunând decât restricții minime fluxului de date.

Funcția de comunicare este presupusă și de unele circuite de prelucrare, cum ar fi cel din figura 9 a cărui conectare în rețea pune probleme esențiale legate de aria pe siliciu ocupată de circuitele de prelucrare și cea ocupată de firele de conexiune pentru care siliciul apare numai ca un suport mecanic [25].

Expandarea în rețea a P din triada P-C-M a făcut ca și termenul median să ia proporții fizice nedorite. Dacă într-o mașină clasică reprezenta o limitare în primul rând conceptuală, într-o structură neconvențională devine o limitare fizică importantă prin *aria ocupată* în rețelele omogene sau prin *circuitele presupuse* în structurile data-flow.

Singura șansă de a minimiza comunicarea este aceea de a reuși și procesări „*la fața locului*” fără a se impune transferul datelor din structura unde sînt memorate. Un prim pas pe această cale îl constituie fragmentarea memoriilor în rețele din ce în ce mai fine, poate la limită în rețele ce în noduri să conțină bitul, într-un context ce stimulează prelucrarea directă.

#### 4.4. Rețele de memorii

Funcția de memorare se manifestă, chiar de la nivelul structurilor de procesare cele mai simple, în două modalități net distincte. Într-un procesor (SO3), format dintr-un A de control și un AS<sup>3</sup> în care are loc prelucrarea, se pune problema *memorării stării*, ca un proces strict temporar fără semnificații perene, sau este nevoie să memorăm o configurație complexă, și cu semnificații ce afectează pe un interval mare de timp funcționarea sistemului, sub forma unei SI. Polarizînd, excesiv poate, memoria poate fi gîndită ca un *tampon temporar* sau ca o *bază de cunoștințe perenă*. În prima calitate servește procese intime într-o mașină digitală, în cea de a doua poate determina funcțiuni de ansamblu. Între aceste două extreme se pot imagina o serie de forme puternic nuanțate. În prima formă este vizată funcția strictă de stocare ce poate foarte bine fi îndeplinită numai de suportul fizic, în cea de a doua formă, memoria presupune și funcții relativ complexe ce conferă o anumită structură *obiectelor*, *înlănțuirii* și *accesului*.

Realizarea în tehnologii VLSI a unor elemente de procesare puternice și a unor configurații de interconectare de mare flexibilitate ce permit un cuplaj ritmic (sistolic), va face ca funcțiile ce se apropie de cea de tampon temporar să fie din ce în ce mai puțin utilizate. La limită, EP din figura 9 poate fi din categoria SO1, nepresupunînd nici funcția de memorare a stării. Ideea dezvoltării în rețea a unor memorii cu posibilități funcționale modeste (exemplu: RAM-uri) nu o vedem ca fiind consistentă cu o aplicație posibilă.

În contrast, structurarea în rețele a memoriilor ce presupun structurarea avansată a obiectelor, înlănțuirii sau a accesului apare ca o alternativă naturală pe calea mării vitezei și degrevării elementelor de procesare de orice efort ce le-ar îndepărta de la mecanismul de evaluare.

Acele mecanisme de acces ce presupun paralelismul se pretează la implementarea unor rețele cît mai „fine” de memorii. Un exemplu tipic în acest sens este oferit de memoria de tip asociativ ce constituie o funcție de



bază în configurarea unei reprezentări complexe asupra realității. Sistemul din figura 8 poate îndeplini această funcție cu atât mai bine cu cât  $n$  este mai mare, oferind astfel șansa ca fiecare  $P_i$  să „caute” într-o memorie de cât mai mici dimensiuni. Notăm faptul că funcțiunile din  $P_i$  nu sînt foarte complexe și, sub o anumită dimensiune a memoriei asociate, pot deveni excesiv de simple.

Conceperea unor mașini orientate către declanșarea unor mecanisme de prelucrare în și pe baze de cunoștințe stimulează conceperea unor rețele de memorii cu funcții asociate relativ complexe [2]. Astfel de mașini baze de date devin adevărate sisteme de procesare complexă în care esențială nu mai este extragerea și introducerea datelor ci în primul rînd definirea unor operații complexe în baza de cunoștințe, rezultatul „citirii” fiind numai reflectat de conținutul memoriei, nu și strict condiționat ca în cazul unei funcții elementare de memorie.

Procese de concurență la nivelul acestor tipuri de rețele sînt, de regulă, foarte mici. Odată cu declanșarea mecanismelor inferențiale „foarte aproape” de rețea se creează însă premisele măririi concurenței.

## 5. Limite ale paralelismului

Paralelismul este stimulat de posibilitățile tehnologice oferite de abordarea VLSI și de conceperea unor algoritmi și a unor stiluri de programare ce-l presupun. De asemenea, el este considerat ca o cale indiscutabilă de creștere a vitezei sistemelor de calcul. În mod cert este o soluție ce va domina gîndirea noastră o bună bucată de timp. Se întrevăd însă, aproape din start, unele limite ce vor fi resimțite mai mult sau mai puțin intens în viitor. Constituie ele germenii ce vor face să apară tendințele de înlăturare a paralelismului, ca fiind limitaiv în gîndirea arhitecturală?

### 5.1. Numeric-nenumeric

Secțiunile 3 și 4 ale acestei lucrări au pus în evidență două moduri net distincte de abordare a unei mașini digitale. Această distincție este condiționată în primul rînd de natura obiectelor procesate în cele două cazuri. În primul (secțiunea 3) erau luate în considerație *șiruri de simboluri* pentru care era deosebit de semnificativă relația dintre elementele șirului. Cel de-al doilea caz (secțiunea 4) presupunea operarea cu *masive de date* în care relațiile dintre elemente erau relativ rigide, dar semnificația pe ansamblu nu era afectată de natura elementelor relaționate.

În cazul recunoașterii sau generării șirurilor de simboluri printr-o gramatică dată, chiar dacă erau create premisele structurale ale paralelismului, acesta nu putea fi declanșat din considerente conceptuale.

Între o *matrice de numere* și un *șir de simboluri* există o diferență esențială ce se poate compara cu *distincția dintre structura informațională și informație*.

Atît legăturile interne, structura sintactică, cît și raportarea la exterior, prin conotații semantice, în cazul numeric și lingvistic, fac în așa fel încît paralelismul să fie presupus aproape natural de aplicațiile numerice și să poată fi declanșat cu restricții foarte mari pentru cele nenumerice.



Nenumericul cînd va fi abordat dintr-o perspectivă avansat paralelă va presupune un paralelism de o natură specială.

Calculatoarele au debutat prin a fi mașini orientate spre numeric, dar trec în etapa actuală printr-o perioadă în care aceste aspecte conceptuale devin marginale. Arhitecturile paralele repun în atenția noastră procesarea numerică, dar numai cu titlu de exemple spectaculoase [19, 25], fără a putea oferi aplicații la fel de elocvente pentru procesarea lingvistică. Entitățile lingvistice sînt preponderent liniar structurate în raport cu semnificațiile asociate, spre deosebire de cele numerice ce devin abordabile și într-o viziune globală, nestingherite de relații complexe între elemente.

Putem concluziona că structurile de procesare propriu-zise, gîndite ca rețele complexe, sînt cu precădere orientate către numeric în cazul în care sînt structurate cu elemente de procesare pure în nodurile rețelei.

## 5.2. LISP — limbaje funcționale

Așa cum am arătat în [20], limbajul LISP nu permite la nivel conceptual declanșarea nestingherită a paralelismelor, acestea putînd apare eventual la nivelul implementării. Realizările actuale de mașini LISP, în variantele cele mai performante chiar, nu au oferit exemple de arhitecturi paralele. Caracterul *nenumeric* al limbajelor de tip LISP nu mai trebuie argumentat.

Spre deosebire de clasa acestor limbaje, cele pur funcționale implică paralelismul aproape ca pe o condiție necesară în implementare. Exemplele tipice [3, 17] de implementare a unor mașini pentru astfel de limbaje nu lasă loc nici unui echivoc. Aplicațiile, curent prezentate ca exemplificînd funcționarea acestor structuri, au o puternică tentă *numerică*.

Acțiunea la nivel de simbol, de element al unei structuri de date, *depinde foarte puternic de context* în primul caz și practic este independentă de acesta în cel de al doilea. Această distincție este esențială în posibilitatea declanșării proceselor paralele.

Trecerea de la numeric la nenumeric se face în contextul tehnologic ce stimulează puternic paralelismul structural, dar în condițiile în care conceptual paralelismul este atrofiat. Cum se poate rezolva această situație paradoxală? Mergînd mai departe pe calea ce pornind de la *aritmetică* trece prin *logică* și va ajunge la *cunoaștere*.

## 5.3. Aritmetică-logică-cunoaștere

Se oferă paralelismului o nouă posibilitate de desfășurare: reprezentarea cunoașterii, cu toate funcțiunile pe care acesta le implică.

Accesul la o structură de date de mare complexitate se poate face în condițiile unui paralelism practic nestingherit de concurență. Comentînd aspectele legate de rețelele de memorii am arătat (secțiunea 4.4) ce formă poate lua paralelismul în configurarea *înlănțuirii* și *accesului* într-o funcție de memorie complexă.

Definirea unor funcțiuni asupra bazelor de cunoștințe necesită ca instrument esențial o bază relațională de date ce se poate afla în dife-



rite raporturi cu o mașină ce realizează procese de inferență și cu una ce configurează o bază de cunoștințe [2].

Se pune problema chiar a realizării unei ierarhii de rețele de memorii ce interacționează, prin rețele complexe de interconectare, între ele și cu elementele unor rețele de procesoare ce aplică diverși operatori.

Trecerea de la pur logic (nenumeric) către operarea *cu și asupra* bazelor de cunoștințe oferă o nouă perspectivă paralelismului. La limită paralelismul se află în situația de a avansa atât de mult încât să degenereze într-un mod de abordare în care *totalul paralelism* să impună un alt concept dominant, o nouă perspectivă în care rolul acestuia să fie atât de implicit încât să dispară în calitate de concept semnificativ. Un rol important în acest proces îl va avea evoluția tehnologiilor VLSI.

Am putea concluziona că există riscul apariției unui „logic bottleneck” prin depășirea numericului prin logic și înaintea definirii mașinilor inferențiale ce folosesc drept resurse reprezentării complexe ale realității. Aspectele conceptuale ale unui „logic bottleneck” sînt foarte importante și net puse în evidență de conceperea unor mașini ce devin performante cu mari dificultăți în absența unui sprijin din partea abordărilor paralele (exemplu: mașinile LISP). Deschiderea pe care o oferă abordarea calculatoarelor din noua generație este mult mai mare decît depășirea restricțiilor impuse de „von Neumann bottleneck”, ajungînd să propună soluții și pentru evitarea limitărilor impuse de „logic bottleneck”.

## 6. Eludarea comunicării

Mecanismele de comunicare într-o mașină digitală au fost și sînt încă foarte importante, chiar dacă rolul acestei funcții a evoluat foarte mult pe măsură ce gîndirea arhitecturală a impus noi soluții. Chiar într-un context dat, comunicarea este condiționată de mai mulți factori.

### 6.1. Comunicarea ca o consecință a incompatibilităților tehnologice

De la începuturi și pînă în zilele noastre accesul la capacități mari de memorie a fost realizat prin intermediul unor memorii ierarhic organizate și limitate de registru și disc la cele două extremități. Spre exemplu, înlănțuirea: disc magnetic — memorie de mare capacitate pe semiconductori — memorie cash de mare viteză — registru, presupune un șir de comunicări între diverse suporturi realizate de regulă în tehnologii net distincte (materiale magnetice, tehnologie planară unipolară, tehnologie planară bipolară). Această ierarhizare nu este impusă conceptual sau funcțional ci vine să compenseze numai limitări tehnologice sau o gîndire sistemică limitativă.

Am amintit deja o altă premisă a comunicării impusă de disjuncția procesor-memorie ce a avut, mai are poate încă, și o motivație tehnologică. Incompatibilitatea primordială între tehnologiile de procesare (relee, tuburi cu vid, dispozitive semiconductoare) și cele de memorare (preponderent magnetice) a sugerat, poate, și a sprijinit demersul formal în sensul mașinilor de tip von Neumann.



## 6.2. Comunicarea ca o consecință a paralelismului

Nodurile unei rețele se află, de regulă, sub imperiul conexiunilor ce afectează chiar natura funcției asociate. Rețelele nu au înlăturat comunicarea, ci gîtuirea introdusă de aceasta, cu prețul creșterii ponderii conexiunilor în efortul de implementare. Strategiile de realizare a interconexiunilor în abordarea structurilor VLSI capătă în unele cazuri o importanță la fel de mare ca cele pentru definirea și realizarea elementelor de procesare [25].

Accesul la operanți sau la resurse se face printr-o rețea de comunicare în cazul unei structuri paralele de procesare. Aspectele legate de relația dintre paralelism și concurență sînt puternic legate de comunicare, conexiuni.

Observăm că *separarea elementelor de procesare de operanți și resurse* este cea care impune funcția de comunicare. În cazul în care s-ar putea realiza o simbioză avansată între aceste trei entități, problema comunicației în structură s-ar pune mai puțin stringent, importantă rămînînd numai problema accesului *la și de la* structură.

## 6.3. Comunicarea ca o consecință a abordării funcționale

Procesarea în noile arhitecturi pune, în esență, problema luării în considerație a unor *mărimi distribuite* și generarea unui rezultat, într-o structură de regulă puternic distribuită.

Într-un prim caz o mărime *din exteriorul sistemului* presupune „confruntarea” cu una internă. Comunicarea este iminentă în acest caz. Dar în cazul în care este vorba de două sau mai multe mărimi *din interiorul sistemului* o configurare convenabilă și o abordare din perspectiva interacțiunii datelor cu hardware-ul asociat strict celulelor ce asigură stocarea lor, pot asigura minimizarea cerințelor în privința comunicării.

Abordarea *funcțională* presupune injectarea într-o rețea a unui flux de date relativ mare, care interacționînd cu structura fizică și cu o, eventual, rudimentară structură informațională, să genereze șirul datelor de ieșire. O abordare din perspectiva *cunoașterii* presupune o situație oarecum inversă : o structură de date relativ mică și simplă structurată este injectată într-un sistem unde interacționează, în principal, cu o structură informațională de mare complexitate. Nu știm în ce măsură o astfel de soluție mai este de tip funcțional.

## 6.4. Conexitate

Evitarea comunicării va presupune două aspecte foarte importante, unul legat de structura fizică iar celălalt de cea informațională.

Realizarea, în primul rînd, a unei *conexiuni*, pînă la identificare, între *procesor și memorie*. La limită acest lucru poate fi perfect realizat asociînd fiecărui bit un element de procesare. În acest caz comunicarea în rețea va mai fi obligatorie, rămînînd să fie evitată cea *la și de la* rețea a elementelor din SI internă. Un prim pas în acest sens, modest deocamdată, îl constituie memoriile asociative realizate hardware.

O a doua problemă, poate mai importantă și mai dificil de realizat, este *caracterul conex al reprezentărilor*, în sensul că este posibilă defi-



nirea adiacenței într-un spațiu multidimensional, astfel încât să se evite pe cât posibil deplasarea în rețea a elementelor structurii informaționale. Îndeplinirea acestor deziderate este legată de progrese notabile atât în tehnologiile VLSI cât și în gândirea modului de reprezentare a datelor. O primă problemă rămîne realizarea compatibilității tehnologice, evitînd astfel limitările expuse în paragraful 6.1. De asemenea, *realizarea conexității într-un spațiu multidimensional* este în egală măsură o problemă tehnologică și una teoretică.

Acest mod de gândire ar duce la omogenizarea foarte avansată a structurilor digitale în așa măsură încît problema *paralelismului* ar fi pusă în umbră de aspectele legate de *conexitate*. Conceperea unor *sisteme tehnologice și conceptual conexe* ar iniția un nou mod de gândire, fundamental deosebit de cele două etape anterioare, cea dominată de arhitectura centralizată, în curs de a fi depășită, și cea centrată în jurul structurilor paralele în curs de a se impune.

## 7. Mașina finită — algoritm

O mașină structurată paralel preia în modul de realizare al interconexiunilor o parte din programul asociat funcției. Există aspecte algoritmice legate sau impuse de rețeaua de procesare. Dacă mașina ar putea fi extinsă oricît, algoritmul ar putea fi preluat în întregime de mașină. În condițiile unei mașini limitate, sub aspect fizic, se pune problema unui reziduu algoritmic ce rămîne exterior, din considerente de finitudine a mașinii sau de optimizare a timpului de execuție.

Spre exemplu: într-o rețea ce realizează produsul matricelor de  $n \times n$  elemente, indiferent de numărul elementelor de procesare, se pot realiza operații pentru  $n$  specificat, oricît de mare, în pofida limitelor fizice, dacă se găsește un algoritm convenabil de a injecta și extrage din rețea elementele matricelor operanți și, respectiv, matricei rezultat. Pe de altă parte, utilizînd o rețea ierarhizată sau data-flow se poate optimiza timpul de execuție printr-o convenabilă configurare a datelor de intrare. Algoritmul de alimentare a mașinilor, în ambele cazuri, este *exterior* structurii fizice.

O mașină ce se dorește limitată fizic și-și propune evaluări în intervale de timp optimizabile va presupune în exteriorul ei, indiferent de gândirea ce a stat la baza ei, un algoritm. Am putut gândi modul de înlăturare a triadei P-C-M, dar în etapa actuală nu putem evita cuplajul MAȘINĂ + ALGORITM ce pare a fi o reeditare a cuplului structură-informație.

Un aspect important este acela că algoritmul are numai rol de *extindere a unei funcții ce aparține mașinii* și nu mai determină exclusiv aspectele funcționale, cum o făcea prin intermediul programelor. Mai nunațat putem spune: MAȘINĂ FINITĂ + ALGORITM DE EXTINDERE.

Dacă SI optimizează structura fizică a procesorului, algoritmul extinde domeniul de utilizare al mașinii. Mașina s-a aliat cu informația, într-o fază incipientă, iar lucrurile au evoluat în sensul că informația a acaparat domeniul extinderilor funcționale. Informația a acționat *intensiv*. Algoritmul asociat unei rețele de procesare paralele are numai un rol *extensiv*, fără veleități în specificarea funcțională. Funcția rămîne la nivelul mașinii.



Prin aceasta saltul făcut de generația a cincea de calculatoare este mult mai mare decât cele anterioare. Există o șansă pentru a scăpa de dogma decuplării funcției de structură prin intermediul programelor, în favoarea unei gândiri ce asociază algoritmului exterior mașinii numai virtuți cantitative.

## BIBLIOGRAFIE

1. AHO, A. V., ULMAN, J. D., *The Theory of Parsing Translations and Compiling*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1973.
2. AMAMIYA, M. ș.a. *New Architecture for Knowledge Base Mechanism*, în *Fifth Generation Computer Systems*, ed. T. MOTO-OKA, North-Holland Publishing Co., 1982.
3. ARVIND, ș.a., *A Processing Element for a Large Multiple Processor Dataflow Machine*, Proc. Int. Conf. on Circuits and Computers, Oct., 1980.
4. ARVIND, GOSTELOW, K. P., *The U — interpreter*, IEEE Computer, Febr., 1982.
5. BACKUS, J., *Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs*, Communications of the ACM, **21**, 8, Aug. (1978).
6. CHOMSKY, N., *A note on phrase structure grammars*, Informations Control, **2** (1959).
7. CHOMSKY, N., *Formal properties of grammars*, în *Handbook of Mathematical Psychology* (2), Wiley, New York (1963).
8. CREANGĂ, I. ș.a., *Introducere algebrică în informatică. Limbaje formale*, Edit. Junimea, Iași, 1974.
9. DENNIS, J. B., *First version of a data flow procedure language*, Techn. Mem. No. 61, Lab. for Computr. Sci., M.I.T., Cambridge, Mass., Mai, 1973.
10. DRĂGĂNESCU, M., *Spre o teorie generală a informației* (preprint), București, Institutul Central pentru Conducere și Informatică, mai, 1983.
11. DRĂGĂNESCU, M., *Semantică și subiect* (preprint), București, Institutul Central pentru Conducere și Informatică, mai, 1983.
12. DRĂGĂNESCU, M., *Considerații filozofico-matematice asupra legăturii dintre formal și neformal* (preprint), București, Institutul Central pentru Conducere și Informatică, mai, 1983.
13. DRĂGĂNESCU, M., *Information, Heuristics, Creation*, în vol. *Artificial Intelligence and Information — Control Systems of Robots*, I. Plander (editor), Elsevier Publishers B. V. (North-Holland), 1984.
14. FEIGENBAUM, E. A., *Innovation and symbol manipulation in fifth generation computer systems*, în *Fifth Generation Computer Systems*, ed. T. MOTO-OKA, North-Holland Publishing Company, 1982.
15. KITSUREGAWA, M. ș.a., *Applications of Hash to Data Base Machine and Its Architecture*, în *New Generation Computing*, **1**, 1 (1983).
16. LEROI-GOURHAN, A., *Gestul și cuvântul*, Edit. Meridiane, București, 1983.
17. MAGO, G. A., *A Network of R Microprocessors to Execute Reduction Languages*, Int. Journ. of Computer and Information Sciences, **8**, 5, și **8**, 6 (1979).
18. MINSKY, M., *Computations. Finite and Infinite Machines*, Prentice-Hall, Inc., 1967.
19. MEAD, C., CONWAY, L., *Introduction to VLSI Systems*, Addison-Wesley Publishing Company, 1980.
20. PĂUN, A. ș.a., *DIALISP — experiment de structurare neconvențională a unei mașini LISP* (în prezentul volum p. 160).
21. ȘTEFAN, G. M. ș.a., *Circuite integrate digitale*, Edit. didactică și pedagogică, București, 1983.
22. ȘTEFAN, G. M., *Structurări neechilibrate în sisteme de prelucrare a informației*, în *Inteligenta artificială și robotica*, Edit. Academiei, 1983.
23. ȘTEFAN, G. M., *Circuite LSI pentru procesoare* (teză de doctorat), Institutul politehnic din București, 1979.
24. UCHIDA, S. ș.a., *Outline of the Personal Sequential Inference Machine: PSI*, în *New Generation Computing*, **1**, 1 (1983).
25. ULLMAN, J. D., *Computational aspects of VLSI*, Computer Science Press, 1984.



# TEHNOLOGII PENTRU CIRCUITE INTEGRATE PE SCARĂ FOARTE MARE

RADU M. BÂRSAN\*)

LARGE-SCALE INTEGRATED CIRCUITS TECHNOLOGY. The paper presents the main features of the evolution of MOS integrated circuits technology, the problems and the major technological trends in this area. The national advancement in microelectronics is also discussed.

## 1. Introducere

Asistăm în zilele noastre la începutul unei epoci de transformări fundamentale, epoca unei noi „revoluții industriale” [1], a unui nou „val” [2], determinat de progrese cu adevărat revoluționare în domeniul electronicii, automaticii, informaticii și telecomunicațiilor. Componenta centrală a bazei acestui progres o constituie evoluția *microelectronicii*, a circuitelor integrate realizate în primul rînd cu ajutorul tehnologiei MOS (Metal-Oxid-Semiconductor).

Pe vremea cînd circuitele integrate nu depășiseră cîteva mii de componente/cip, s-a încetățenit clasificarea acestora în circuite integrate pe scară mică (SSI — Small Scale Integration) avînd pînă la 100 componente/cip, pe scară medie (MSI — Medium Scale Integration) cu pînă la 1000 componente/cip, pe scară mare (LSI — Large Scale Integration) cu pînă la 10 000 componente/cip și pe scară foarte mare (VLSI — Very Large Scale Integration) cu pînă la 100 000 componente/cip. Cînd au fost realizate circuite cuprinzînd multe sute de mii de componente, epuizîndu-se gradele de comparație, s-a vorbit de integrarea pe scară „ultra” mare (ULSI).

Creșterea complexității circuitelor integrate a avut ca efect imediat scăderea dramatică a costului și creșterea performanțelor sistemelor clasice de prelucrare a informației. Pe de altă parte însă, oferind posibilitatea integrării foarte aproape unele de altele a elementelor de procesare și a celor de memorie, ea face posibil un salt calitativ: atingerea unui grad înalt de prelucrare simultană (paralelă) a informației [3]. Vom atribui termenul de integrare pe scară *foarte mare* (VLSI), complexității circuitelor care depășesc acest prag revoluționar.

Nu ne putem pune problema abordării detaliate într-o lucrare de dimensiuni atît de reduse a tehnologiilor dezvoltate în scopul realizării de circuite MOS/VLSI. Vom încerca însă să prezentăm cîteva aspecte principale care jalonează evoluția tehnologiei circuitelor integrate MOS, problemele care stau în fața integrării pe scară foarte mare și unele considerații privind progresele înregistrate în țara noastră în domeniul tehnologiilor pentru microelectronică.

\*) Intreprinderea Microelectronica.



## 2. Componentele creșterii complexității circuitelor integrate MOS

De la primele porți logice realizate cu linii de  $25\text{ }\mu\text{m}$  până la microprocesoare cu peste 450 000 de tranzistoare realizate cu linii de  $1,5\text{ }\mu\text{m}$ , au trecut mai puțin de două decenii. Această creștere rapidă a complexității circuitelor integrate, ilustrată în figura 1, a fost rezultatul a trei contribuții majore.

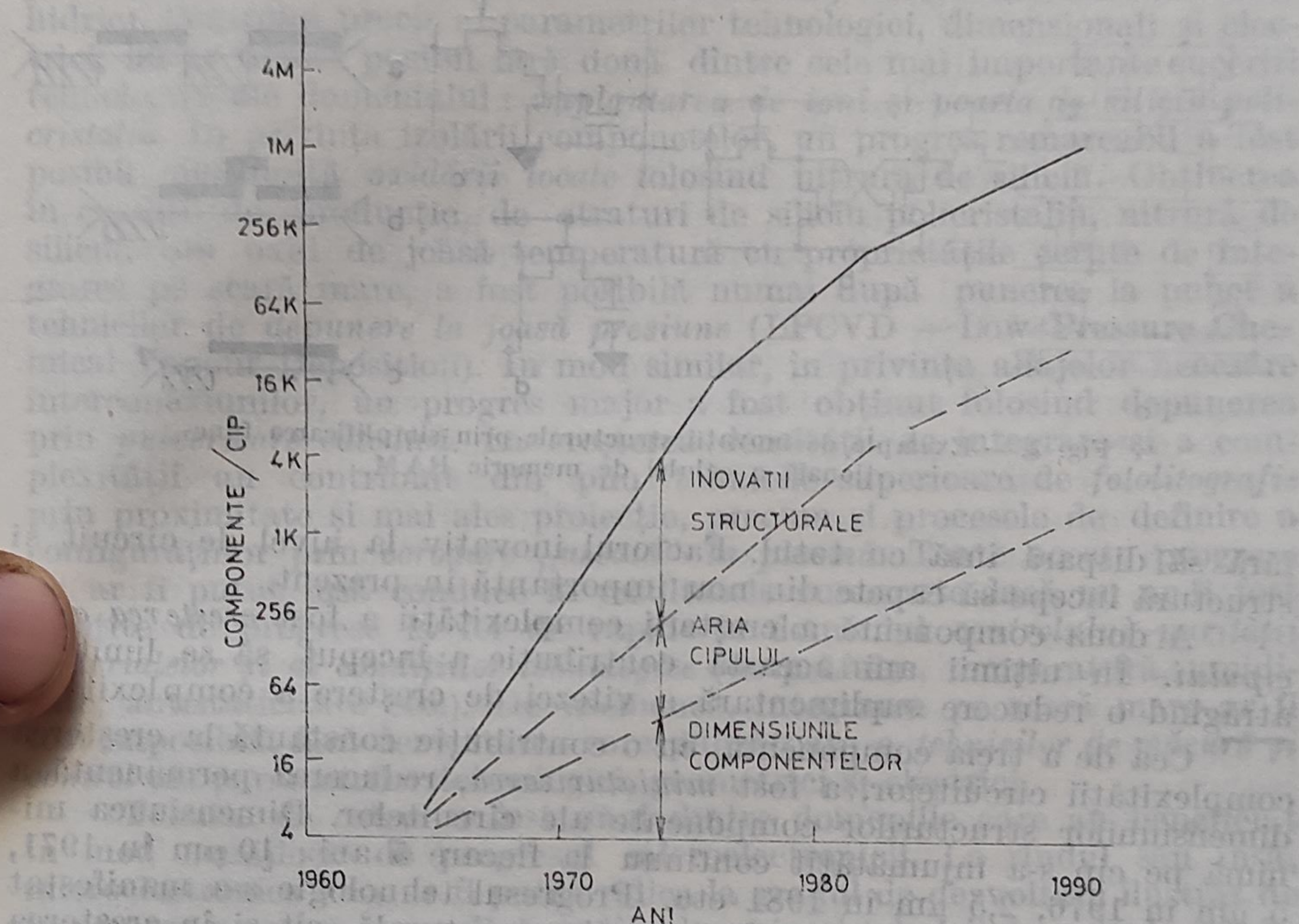


Fig. 1. — Componentele creșterii complexității circuitelor integrate MOS.

O primă componentă o constituie *inovația structurală*, atât la nivel de circuit cât și de dispozitiv. În figura 2 sînt exemplificate astfel de inovații, în cazul celulelor de memorii cu acces aleator (RAM). Pe direcția simplificării circuitelor (A) s-a trecut de la celula statică cu 8 tranzistoare (apoi 6 tranzistoare) (a), la celula dinamică cu 4 tranzistoare (b), urmată de celula cu 3 tranzistoare (c) și, în fine, cu un singur tranzistor (d). În cazul registrelor de deplasare, s-a trecut de la registrele statice cu bistabile cu 16 tranzistoare/celulă, la registre dinamice fără raport cu 6 tranzistoare/celulă și în fine la registre cu transfer de sarcină cu echivalentul a numai două tranzistoare/celulă.

Pe de altă parte, însăși structura componentelor a suferit modificări. Pentru exemplificare, în figura 2B este ilustrată evoluția celulei dinamice „cu un tranzistor”: (a) varianta inițială în care capacitorul și tranzistorul sînt discernabile, (b) structura rezultată prin alipirea electrozilor de stocare și transfer și (c) forma obținută prin contopirea acestor electrozi și definirea zonei de stocare a celulei prin implantare



de ioni. Se remarcă faptul că această evoluție structurală a condus la pierderea identității componentelor elementare clasice (tranzistor, capacitor etc.) și contopirea acestora în *structuri funcționale* noi [4]. În a doua jumătate a deceniului trecut, inovația structurală a încetat să mai constituie o contribuție importantă la creșterea complexității circuitelor,

#### A. Simplificarea circuitelor

#### B. Simplificarea structurilor

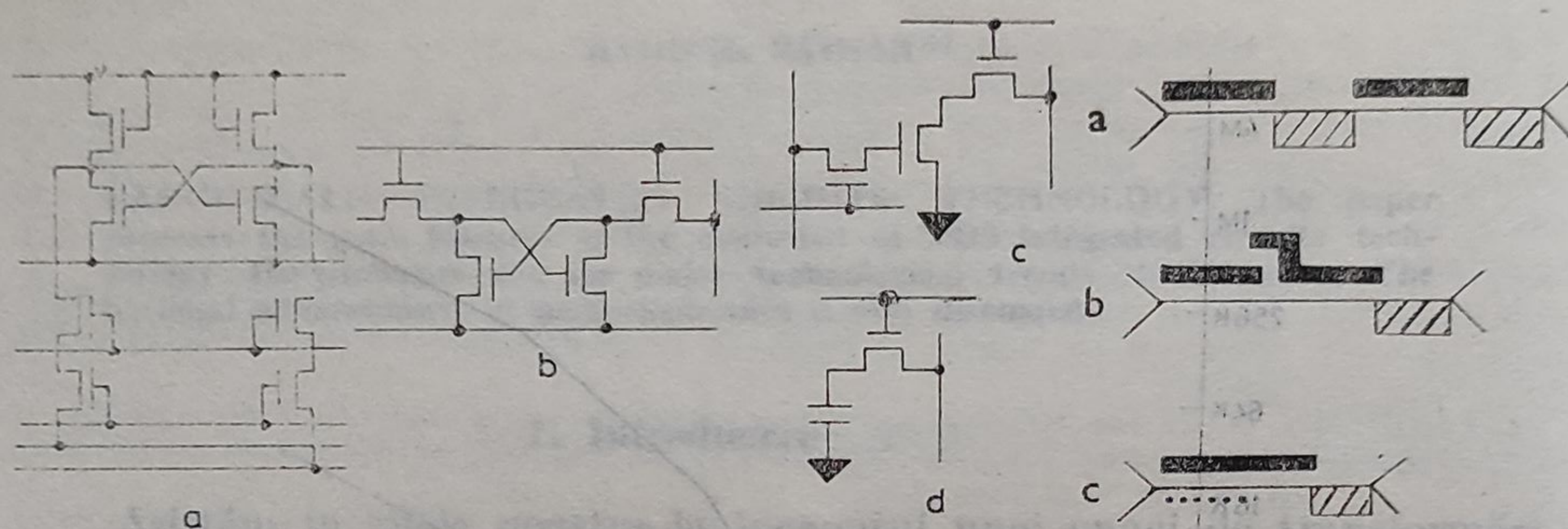


Fig. 2. — Exemple de inovații structurale prin simplificarea funcțională a celulei de memorie RAM.

fără să dispară însă cu totul. Factorul inovativ la nivel de circuit și structură începe să capete din nou importanță în prezent.

A doua componentă a creșterii complexității a fost *creșterea ariei cipului*. În ultimii ani această contribuție a început să se limiteze, atrăgând o reducere suplimentară a vitezei de creștere a complexității.

Cea de a treia componentă, cu o contribuție constantă la creșterea complexității circuitelor, a fost *miniaturizarea*, reducerea permanentă a dimensiunilor structurilor componente ale circuitelor. Dimensiunea minimă pe cip s-a înjumătățit continuu la fiecare 5 ani:  $10\ \mu\text{m}$  în 1971,  $5\ \mu\text{m}$  în 1976,  $2,5\ \mu\text{m}$  în 1981 etc. Progresul tehnologic s-a manifestat din plin atât în componenta de inovație structurală, cât și în creșterea ariei cipurilor, dar rezultatul cel mai direct a fost creșterea densității de integrare prin reducerea dimensiunilor componentelor. Aceasta a făcut ca dimensiunea minimă pe cip să fie considerată în mod convențional o primă măsură a evoluției tehnologiilor circuitelor integrate MOS.

În privința dimensiunii minime pe cip, evoluția nu a fost și nu va fi nici în viitor continuă. Discontinuități au apărut la trecerea de la  $10\ \mu\text{m}$  la  $5\ \mu\text{m}$  și de la  $5\ \mu\text{m}$  la  $2,5\ \mu\text{m}$ . Următoarele bariere ce vor trebui depășite pentru producerea de circuite integrate pe scară foarte mare se află în jurul valorilor de  $1,3\ \mu\text{m}$  și  $0,5\ \mu\text{m}$ . Depășirea acestor bariere presupune în primul rând modificări structurale și tehnologice majore, așa cum a fost spre exemplu saltul de la tehnologiile cu poartă de aluminiu la tehnologiile cu poartă de siliciu. În al doilea rând, discontinuitățile apar datorită faptului că dincolo de anumite limite se fac simțite fenomene fizice complet noi care necesită modele adecvate și timp pentru a fi înțelese și controlate.

Cu toate realizările deosebit de spectaculoase în domeniul tehnologiilor MOS, ne aflăm încă departe de limitele previzibile [5]. Astfel, *densitatea maximă de integrare*, corespunzătoare unei dimensiuni minime pe cip de  $0,2\ \mu\text{m}$ , este estimată la circa  $10^8$  componente/ $\text{cm}^2$ , de aproxima-



tiv 200 de ori mai mare decât în prezent (1984), iar *complexitatea* (numărul de componente/cip) poate crește teoretic cu peste 3 ordine de mărime față de realizările actuale.

Fără a intra în detalii, trebuie totuși să menționăm pe scurt câteva din procesele tehnologice care au avut efecte majore până în prezent asupra tehnologiilor pentru circuite integrate MOS, făcând posibilă integrarea pe scară mare în producție de serie. În domeniul stabilității parametrilor structurii MOS, un efect important l-a avut *oxidarea în prezența clorului* (sub formă de tricloretilenă, triclorețan sau acid clorhidric). Controlul precis al parametrilor tehnologici, dimensionali și electrici, nu ar fi fost posibil fără două dintre cele mai importante cuceriri tehnologice ale domeniului: *implantarea de ioni* și *poarta de siliciu policristalin*. În privința izolării componetelor, un progres remarcabil a fost posibil mulțumită *oxidării locale* folosind nitrura de siliciu. Obținerea în condiții de producție de straturi de siliciu policristalin, nitrură de siliciu, sau oxid de joasă temperatură cu proprietățile cerute de integrarea pe scară mare, a fost posibilă numai după punerea la punct a tehnicilor de *depunere la joasă presiune* (LPCVD — Low Pressure Chemical Vapour Deposition). În mod similar, în privința aliajelor necesare interconexiunilor, un progres major a fost obținut folosind depunerea prin *pulverizare catodică*. La creșterea densității de integrare și a complexității au contribuit din plin tehnicile superioare de *fotolitografie* prin proximitate și mai ales proiecție, precum și procesele de definire a configurațiilor prin *corodare „uscată” în plasmă*. Toate aceste progrese nu ar fi putut însă conduce la rezultatele cunoscute dacă nu ar fi fost însoțite de progrese la fel de rapide în domeniul *controlului purității materialelor și al condițiilor tehnologice* (desprăfuire, temperatură, umiditate, antistaticizare etc.). De asemenea, integrarea pe scară mare ar fi fost imposibilă fără dezvoltarea corespunzătoare a *tehnicilor de măsură și control* ale parametrilor tehnologici, geometrici și electrici.

Tehnica de calcul a fost unul dintre domeniile care au beneficiat în mod nemijlocit de progresul microelectronicii. La rîndul său însă, microelectronica nu s-ar fi putut ridica la gradul de dezvoltare ilustrat de integrarea pe scară mare, fără sprijinul masiv al tehnicii de calcul, atât în cadrul proiectării și testării asistate de calculator a circuitelor, cât și în cadrul automatizării proceselor tehnologice.

### 3. Problemele integrării pe scară foarte mare

Realizarea de circuite integrate pe scară foarte mare întâmpină o serie de dificultăți, pe care vom încerca să le enumerăm mai jos. O primă categorie de probleme o constituie *fenomenele fizice diferite* care intervin în cazul structurilor ultraminiaturizate [5]. Spre exemplu, datorită conducției sub prag și a fluctuațiilor statistice ale atomilor de impuritate în canalul tranzistoarelor, tensiunea de alimentare a circuitelor nu va putea fi redusă sub aproximativ 1 V. În consecință, structurile nu vor putea respecta principiul reducerii proporționale, lucrînd cu cîmpuri electrice interne apropiate de cîmpul critic. În aceste condiții, efectele fizice asociate cîmpurilor intense vor avea o pondere deosebit de mare. De asemenea, efectele bidimensionale devin semnificative în cazul structurilor ultrami-



niaturizate, fapt care conduce la necesitatea revizuirii majorității modelelor clasice.

În circuitele VLSI *interconexiunile* reprezintă o altă problemă de mare importanță. Un prim motiv este acela că aria ocupată de traseele de interconexiuni crește vertiginos odată cu creșterea complexității circuitelor, fiind cel puțin egală cu aria ocupată de elementele active. În consecință, obținerea de trasee conductive cât mai fine, situate pe cât mai multe niveluri izolate între ele, va fi un desiderat permanent. O altă categorie de motive pentru care va fi nevoie să se acorde o atenție crescândă interconexiunilor, constă în limitările de viteză asociate propagării semnalelor în circuit, fie că aceste limitări sînt datorate unor fenomene ca electromigrarea, căderii de tensiune pe linii, sau pur și simplu întîrzierilor de propagare.

Astfel de probleme de natură electrică sînt strîns legate de problemele de natură termodinamică, dintre care cea mai serioasă este *disipația finită a căldurii*. Găsirea de sisteme cât mai comode și eficiente de răcire a circuitelor în timpul funcționării va constitui o problemă cheie a realizării de sisteme VLSI.

Problemele geometrice, fizice, electrice și termodinamice nu sînt independente, ci se condiționează reciproc. La nivelul integrării pe scară foarte mare, efectul principal al acestor interacțiuni constă în *imposibilitatea creșterii simultane a complexității și a vitezei de lucru a circuitelor* [5].

Cea de a doua categorie importantă de probleme ce stau în fața integrării pe scară foarte mare este de natură pur tehnologică. Domeniul VLSI nu va putea să se dezvolte fără progrese substanțiale în domeniul *fotolitografiei* și a tehnicilor de *definire a configurațiilor*. *Izolarea componentelor* în cadrul circuitului constituie o dificultate din ce în ce mai mare în calea miniaturizării și nu are în prezent soluții clare de rezolvare. *Redistribuirea impurităților* în timpul proceselor termice rămîne o problemă majoră, în ciuda eforturilor depuse pentru reducerea temperaturilor de lucru și utilizarea exclusivă a implantării de ioni pentru dopare cu impurități.

În cadrul tehnologiilor VLSI *abaterile parametrilor de la valorile nominale* vor constitui în cele din urmă dificultatea principală [5]. De asemenea, *controlul purității materialelor și al condițiilor tehnologice* va determina în mare măsură realizabilitatea circuitelor integrate pe scară foarte mare. În fine, nu trebuie uitat nici faptul că una din problemele practice de mare importanță rămîne *costul*. O tehnologie superioară sau un echipament care permite realizarea unor circuite mai complexe, nu pot fi introduse în fabricație decît în măsura în care creșterea performanțelor circuitelor astfel obținute echilibrează costurile suplimentare, sau, dacă la performanțe egale se obțin sisteme mai ieftine.

Dificultățile tehnice menționate mai sus rezultă din necesitatea procesării cu un grad de control din ce în ce mai mare, a unor plachete din ce în ce mai mari, pe care se găsesc structuri cu o densitate de integrare din ce în ce mai mare. Încă de pe acum încep să apară procese tehnologice, tehnici și echipamente noi, care să facă față acestor cerințe. Vom enumera în continuare cîteva dintre acestea.

Generarea de configurații micronice și submicronice presupune în primul rînd un proces de *fotolitografie* deosebit de rafinat. Sistemele actuale prin proiecție nu vor putea fi utilizate în cazul integrării pe scară foarte mare, atît datorită rezoluției nesatisfăcătoare, cît și datorită sensibi-



lității la deformarea suferită de plachetă în cadrul proceselor termice. Repetarea directă 10 :1 și sistemele bazate pe raze X sau fascicol de electroni vor rămâne singurii candidați pentru VLSI.

În privința corodării diverselor straturi crescute sau depuse pe suprafața plachetei, se vor putea utiliza numai *tehnici anizotrope*, cum ar fi corodarea cu fascicol de ioni reactivi. Pentru dimensiuni submicronice este probabil că vor fi utilizate în locul corodării, tehnici indirecte de definire a configurațiilor cum ar fi lift-off, sau alte metode prin care lățimea unei linii e definită de grosimea unui strat și nu de o mască clasică.

Pentru integrarea pe scară foarte mare vor putea fi utilizate numai procese termice la *temperaturi joase*, de pînă la 700°C, datorită necesității de a avea redistribuții cît mai mici ale impurităților, reducerii la minimum a deformării plachetelor și reducerii densității de defecte. Astfel de procese sînt : oxidarea la presiune înaltă (HIPOX), depunerea în plasmă la joasă presiune (PECVD) și diferite procese termice bazate pe fascicule energetice de tip laser, electronice, sau ionice.

*Materialde* de bază vor suferi adaptări și modificări. În privința substratelor vor cîștiga teren materialele dielectrice cum sînt safirul, sau oxidul îngropat obținut prin implantare cu ioni de oxigen. Siliciurile metalice și metalele refractare vor fi folosite aproape exclusiv, pentru realizarea traseelor conductoare. În privința izolatorilor, se va extinde probabil utilizarea oxinitrurilor.

În fine, dar nicidecum în ultimul rînd, domeniul VLSI va însemna o revoluție în *tehnicele de asamblare a cipurilor*, capsulele clasice fiind abandonate. În prezent se experimentează la scară industrială mai multe soluții de montaj direct al structurilor pe plăci modulare. Un exemplu este sistemul utilizat de IBM pentru calculatoarele 3081, 3083 și 3084, în care 133 cipuri sînt conectate direct pe un substrat ceramic, un mare număr de astfel de substrate fiind asamblate sub atmosferă de heliu într-un modul răcit cu apă. Un astfel de modul conține pînă la 45 000 de cipuri și are 1 800 de terminale. Calculatorul IBM3081 cuprinde 26 de module montate pe patru plăci de 60 × 70 cm, totalizînd circa 800 000 de circuite. Un alt exemplu este sistemul utilizat de Hewlett-Packard pentru calculatorul de 32 de biți HP9000, care cuprinde 2,5 megaocteți de memorie RAM, avînd dimensiunile unui calculator de birou. Tehnica de montaj folosită este similară cablajelor imprimate, utilizînd substrate speciale, în locul circuitelor încapsulate conectîndu-se însă direct cipurile.

Domeniul VLSI va fi în plină dezvoltare către sfîrșitul deceniului. Zorile acestei noi ere în microelectronică au apărut însă de pe acum, materializate în diverse circuite MOS care, cel puțin din punctul de vedere al complexității, prefigurează integrarea pe scară foarte mare. Tabelul 1 prezintă succint cîteva asemenea realizări. În domeniul *memoriilor*, trebuie menționate memoriile RAM dinamice de 256 kbiți realizate atît de firme japoneze cît și americane. În al doilea rînd, noua generație de *procesoare de 32 biți* poate fi socotită ca deschizînd și ea domeniul VLSI. Printre acestea amintim circuitul iAPX432 programabil în limbajul Ada, anunțat de Intel încă din 1981, primul microprocesor de 32 biți capabil să lucreze cu memorie virtuală, NS16032, produs de National Semiconductor, microprocesoarele Motorola 68010 și 68020, TMS 320 produs de Texas Instruments, primul microprocesor de 32 biți realizat în tehnologie CMOS, Bellmac-32, produs de Bell Laboratories și unitatea centrală a calcula-



# REALIZARI CARE DESCHID DOMENIUL VLSI

Circuit	Firma	Caracteristici
Memorii	Toshiba	linii 1,3-2,5 $\mu\text{m}$
DRAM	Hitachi	aria celulei 70-100 $\mu\text{m}^2$
256 K	Motorola	aria cipului 40-50 $\text{mm}^2$
	Texas Instruments	
	Mitsubishi	
	IBM	
iAPX 432	Intel	32 biți, Ada
NS 16032	National Semiconductor	32 biți, memorie virtuală
M 68010/20	Motorola	32 biți
TMS 320	Texas Instruments	32 biți, arhitectura Harvard
BELLMAC 32	Bell	32 biți, CMOS
HP 9000 CPU	Hewlett Packard	32 biți, structura regulată

torului HP9000, realizat de Hewlett-Packard, circuit care conține circa 450 000 tranzistoare și poate prelucra 1 MIPS (milion de instrucțiuni pe secundă).

## 4. Progresul microelectronicii în țara noastră [6]

Tehnologii pentru circuite integrate MOS au început să fie puse la punct pe plan mondial, începînd cu mijlocul deceniului '60. De atunci, viteza de dezvoltare tehnologică a cunoscut o creștere continuă, creștere sprijinită de multe alte industrii (producătoare de echipamente, materiale etc.), precum și de învățămînt, cercetare, domenii care au ținut permanent pasul cu dinamica microelectronicii.

În țara noastră au început să existe preocupări în domeniul tehnologiei MOS, începînd cu anul 1970, la CCPOE-Băneasa, devenit apoi ICCE-București și ulterior CCSITS. Acesta a fost timp de peste 10 ani singurul loc din țară unde s-au făcut tentative de realizare de dispozitive și apoi circuite MOS, în paralel cu un mare număr de cercetări privind dispozitivele semiconductoare, de la tranzistoare bipolare și dispozitive optoelectronice la dispozitive pentru microunde și circuite hibride.

Abia în anul 1981, cînd în S.U.A. intra în fabricație memoria de 64 kbiți, sau cînd în R.D.G. totalul angajaților unei singure fabrici de circuite integrate MOS depășea 5 000, a fost demarat un program de dezvoltare tehnologică la ICCE, pe o linie pilot. În 1983, după mai mult de 7 ani de la începerea investiției, a intrat în funcțiune cercetarea tehnologică și fabricația de circuite integrate MOS la întreprinderea Microelectronica-București.

Pentru ilustrarea dinamicii dezvoltării tehnologiei MOS în țara noastră, figura 3 prezintă evoluția complexității tehnologice a circuitelor realizate prin efort propriu. *Gradul de complexitate al unei tehnologii, este*



măsurat printr-o cifră de merit („ct”), definită ca produsul între complexitatea circuitelor (componente/cip) și numărul de etape tehnologice de dificultate mare, care presupun echipamente și condiții speciale, precum și un efort deosebit pentru punere la punct și menținere în parametri.

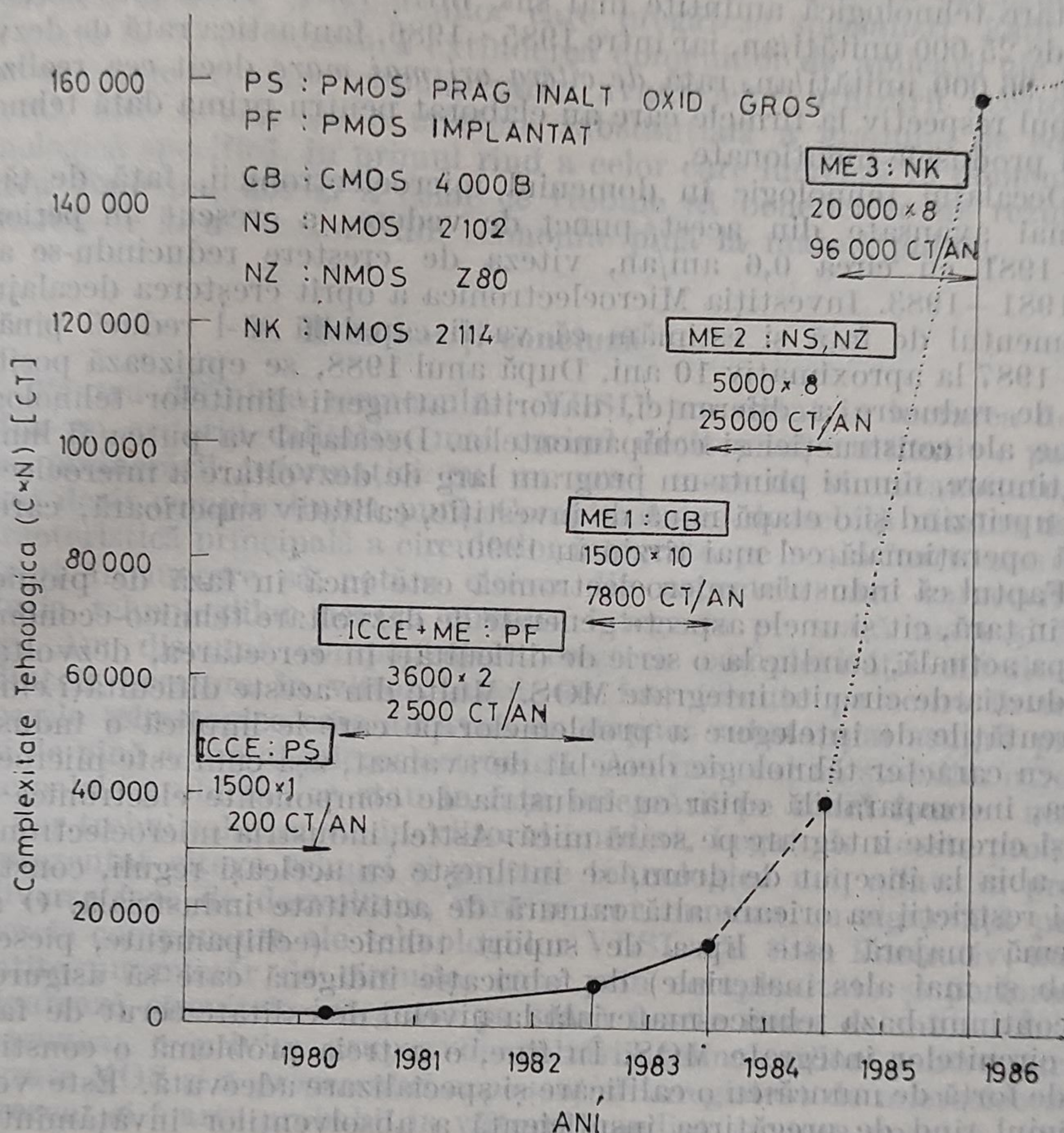


Fig. 3. — Dinamica dezvoltării tehnologiei circuitelor integrate MOS în țara noastră.

Nivelul de complexitate astfel definit este de aproximativ 1 500 unități pentru tehnologia PMOS standard cu prag înalt și oxid de câmp gros, realizată la ICCE până în 1981, 7 200 unități pentru tehnologiile PMOS cu implantări de câmp și pentru controlul tensiunilor de prag, puse la punct pe linia pilot din ICCE, 15 000 unități pentru tehnologiile CMOS cu poartă de aluminiu, realizate la Microelectronica cu dotare parțială (denumită în figura 3 ME1), 40 000 unități pentru tehnologiile NMOS cu poartă de siliciu necesare fabricării memorii de 1 kbit statică și 4 kbiți dinamică și microprocesorului Z80, tehnologii ce vor fi realizate la Microelectronica după completarea dotării (ME2) și circa 160 000 unități pentru tehnologiile NMOS necesare fabricării memorii de 4 kbiți statică și 16 kbiți dinamică și microprocesorului Z8000, ce vor putea fi realizate după achiziționarea noilor echipamente și materiale necesare (ME3).



Figura 3 prezintă și valorile *ratei de dezvoltare* tehnologică în diferite etape. Aceasta a fost de 200 unități/an în perioada „IOCE”, 2 500 unități/an în cadrul programului IOCE-Microelectronica și 7 800 unități/an între 1983—1984 la Microelectronica. Pentru atingerea obiectivelor de dezvoltare tehnologică amintite mai sus, între 1984—1985 este necesară o rată de 25 000 unități/an, iar între 1985—1986, fantastica rată de dezvoltare de 96 000 unități/an, rată *de câteva ori mai mare* decât cea realizată la timpul respectiv la firmele care au elaborat pentru prima dată tehnologiile și produsele menționate.

Decalajul tehnologic în domeniul microelectronicii, față de țările cele mai avansate din acest punct de vedere, a crescut în perioada 1970—1981 cu circa 0,6 ani/an, viteza de creștere reducându-se apoi între 1981—1983. Investiția Microelectronica a oprit creșterea decalajului în momentul de față și estimăm că va fi capabilă să-l reducă până în 1986—1987 la aproximativ 10 ani. După anul 1988, se epuizează posibilitățile de reducere a diferenței, datorită atingerii limitelor tehnologice maxime ale construcției și echipamentelor. Decalajul va putea fi limitat în continuare, numai printr-un program larg de dezvoltare a microelectronicii, cuprinzând și o etapă nouă de investiție, calitativ superioară, care să devină operațională cel mai târziu în 1990.

Faptul că industria microelectronică este încă în fază de pionierat la noi în țară, cât și unele aspecte generale de dezvoltare tehnico-economică în etapa actuală, conduc la o serie de dificultăți în cercetarea, dezvoltarea și producția de circuite integrate MOS. Multe din aceste dificultăți emană din greutățile de înțelegere a problemelor pe care le implică o industrie nouă, cu caracter tehnologic deosebit de avansat, așa cum este microelectronica, incomparabilă chiar cu industria de componente electronice discrete și circuite integrate pe scară mică. Astfel, industria microelectronică, aflată abia la început de drum, se întâlnește cu aceleași reguli, constrângeri și restricții ca oricare altă ramură de activitate industrială. O altă problemă majoră este lipsa de suport tehnic (echipamente, piese de schimb și mai ales materiale) de fabricație indigenă care să asigure în mod continuu baza tehnico-materială la nivelul de calitate cerut de fabricația circuitelor integrate MOS. În fine, o a treia problemă o constituie lipsa de forță de muncă cu o calificare și specializare adecvată. Este vorba în primul rând de pregătirea insuficientă a absolvenților învățământului superior în probleme de tehnologie, datorită lipsei unei dotări a instituțiilor respective cu baza materială necesară, precum și de insuficienta calificare și educație tehnologică a operatorilor ce sînt destinați acestei industrii de vîrf.

În condițiile arătate, dezvoltarea pe baze solide a microelectronicii în țara noastră, nu va putea fi realizată decât printr-un efort mai amplu și mai general decât pînă în prezent, printr-un *program național de dezvoltare* bine încadrat în elementele politicii industriale. La baza acestui program trebuie să stea desigur recunoașterea necesității dezvoltării rapide a domeniului, cu toate implicațiile sale asupra întregii dezvoltări tehnico-economice a țării în viitorul apropiat. Scopurile și strategiile de urmat pentru a obține această dezvoltare trebuie definite clar de la începutul programului. Vor fi inevitabile anumite opțiuni privind specializarea tehnologică, pentru a evita dispersarea eforturilor și a resurselor în prea multe direcții, cum ar fi abordarea prea multor tehnologii diferite.

Așa cum am mai arătat, sînt necesare investiții noi în domeniu, pentru a putea menține relativ constantă diferența față de nivelul mondial.



Dar investițiile izolate nu sînt suficiente. Microelectronica este o industrie de vîrf care nu se poate sprijini decît pe alte vîrfuri industriale (din chimie, mecanică fină, optică, electronică etc.). De aceea efortul de dezvoltare al microelectronicii trebuie corelat cu cel de dezvoltare al ramurilor care o sprijină, inclusiv a domeniilor care pregătesc specialiști. Este pe de altă parte la fel de necesară extinderea domeniilor de aplicații ale produselor microelectronicii, popularizarea avantajelor utilizării acestora. În general, este nevoie de o creștere substanțială a gradului de educație tehnologică specifică, în primul rînd a celor care lucrează în domeniu sau pentru domeniu, dar și a celor ce trebuie să beneficieze de rezultatele obținute, de la utilizatorii din economie pînă la marele public.

## 5. Rezumat și concluzii

Pentru definirea termenului „VLSI” am optat în lucrarea de față pentru un criteriu calitativ, mai curînd decît unul cantitativ: posibilitatea prelucrării informației cu un grad mare de simultaneitate, mai curînd decît complexitatea pură. Complexitatea deosebită rămîne desigur o caracteristică principală a circuitelor integrate pe scară foarte mare. Am încercat în lucrare să notăm cîteva din aspectele care caracterizează evoluția tehnologiilor pentru circuite integrate pe scară mare și foarte mare. Am discutat componentele creșterii complexității circuitelor și tendințele acestora în viitor. Apoi, au fost trecute pe scurt în revistă procesele tehnologice care au avut un impact substanțial asupra progresului de pînă acum al microelectronicii. Am încercat să stabilim care sînt principalele probleme ce stau în fața integrării pe scară foarte mare și care vor trebui soluționate în viitorul imediat. Legat de aceste probleme, am prezentat cîteva tehnici și procese tehnologice noi, aflate în prezent în diverse faze de dezvoltare, care se vor număra cu siguranță printre procesele componente ale tehnologiilor VLSI. Cu titlu ilustrativ, au fost amintite un număr de circuite, care prin complexitate și performanțe prefigurează circuitele integrate pe scară foarte mare de mîine. În fine, am aruncat o privire asupra dezvoltării tehnologiilor pentru circuite integrate MOS și a *progresului exponențial* înregistrat de microelectronică în prezent și foarte probabil în viitorul imediat, în țara noastră. Concluzia analizei este însă aceea că, stabilirea domeniului pe baze solide și păstrarea ratei de progres amintite, nu va putea avea loc fără o atitudine mai activă, manifestată prin amorsarea unui *program național special de dezvoltare a microelectronicii*, în sens mai larg și mai profund decît pînă în prezent. Numai astfel va fi posibilă implementarea în țară a tehnologiilor VLSI, în accepțiunea lucrării de față.

## BIBLIOGRAFIE

1. M. DRĂGĂNESCU, *A doua revoluție industrială, Microelectronica, automatica, informatica — factori determinanți*, Edit. tehnică, 1980.
2. A. TOFFLER, *Al treilea val*, Edit. politică, 1983.
3. C. MEAD și L. CONWAY, *Introduction to VLSI Systems*, Ed. Addison Wesley, 1980.
4. R. BÂRSAN, *Dispozitive și circuite integrate cu transfer de sarcină*, Edit. tehnică, 1981.
5. R. BÂRSAN, *Limite ale circuitelor MOS integrate pe scară foarte mare*, în volumul „Microelectronica” ed. M. Drăgănescu și D. Dascălu, Ed. Academiei (în curs de apariție).
6. R. BÂRSAN, *Asupra unor aspecte specifice ale cercetării și producției de structuri pentru circuite integrate MOS/LSI*, raport intern I, Microelectronica, febr. 1983.



## GENERAȚIA A CINCEA DE CALCULATOARE — UN PAS SPRE PROGRAMAREA NATURALĂ?

CRISTIAN GIUMALE\*)

„Visele nu trebuie crezute. Ele trebuie, ca și vinurile bune, degustate, niciodată cu alt gând în cap decât acela al soiului de viță de vie din care au fost trase”.

NICHITA STĂNESCU

**FIFTH-GENERATION COMPUTERS—A STEP TOWARDS NATURAL PROGRAMMING?** The impact of the new computer generation on programming is briefly discussed. The trend towards a more natural way of thinking about programs is illustrated by several examples taken from the field of deductive problem solving. LISP and a PROLOG-like system were used as programming tools.

### 1. Introducere

Rezolvarea oricărei probleme, cu calculatorul, mai presupune încă elaborarea unui algoritm detaliat, codificabil într-un limbaj de programare. De aici derivă două puncte de vedere, considerate multă vreme fundamentale, în ceea ce privește perspectiva evoluției modului de programare :

a) Se speră că nivelul general de educație al populației utilizatorilor potențiali va crește, marea lor majoritate fiind familiarizată cu cel puțin un limbaj de programare de nivel înalt.

b) Se speră în apariția unui limbaj de programare cât mai general și ușor de utilizat.

Din nefericire, compromisul dintre generalitate și eficiență conduce la complexitate, iar încercări recente, cum ar fi bunăoară ADA, au dovedit acest fapt. Pe de altă parte, este greu de presupus că marea masă a utilizatorilor va adopta în scurtă vreme un astfel de limbaj. De fapt, aceste două puncte de vedere nu țin seama de progresele tehnologice remarcabile survenite în ultima vreme în ceea ce privește realizarea arhitecturilor hard specializate și, mai ales, nu consideră importante mutațiile survenite în natura problemelor rezolvate cu calculatorul.

\*) Institutul politehnic din București.



## 2. O perspectivă a programării

Descentralizarea puterii de calcul și integrarea tot mai profundă a calculatorului în cotidian nu mai poate fi concepută fără o „umanizare” a modului de dialog, fără o schimbare a opticii față de modul de programare. Din acest punct de vedere, generația a cincea de calculatoare, de care se vorbește atât de mult, este în primul rând un manifest al apropierii calculatorului de om [1]. Trebuie să admitem că toate progresele soft realizate au vizat, mai ales, performanțe legate de costul utilizării echipamentelor, interfața om-mașină fiind deseori sacrificată, iar această interfață nu trebuie judecată doar prin prisma unui limbaj de comandă sau de programare ci, mai larg, prin însăși modul de a gândi programarea calculatoarelor și, mai fundamental, prin modul de a rezolva problemele.

În esență este vorba de structura și nivelul algoritmului de rezolvare și de modul de specificare al acestuia. Idealul este clar doar din punctul de vedere al limbajului, dorit natural. În ceea ce privește modul de rezolvare, problema este deschisă cercetărilor. Am dori un calculator care să poată fi programat așa cum gândim, dar oare cum gândim? Care sînt acele primitive care ar permite reconstrucția oricărui proces de gândire și ce structură ar avea această construcție?

Proiectul japonez, de care deseori este asociată generația a cincea de calculatoare, probabil din admirație, a optat deja pentru programarea logică, tipică proceselor de raționament deductiv. Este doar un pas mic, dacă avem în vedere cît de diverse sînt problemele cu care sîntem confrunțați în mod obișnuit și cît de divers le rezolvăm, prin inducție, analogie, deducție, intuiție și, deseori, din întâmplare. Dar marșul de 1 000 de mile a început.

Cîștigul esențial al noilor generații de calculatoare nu este, cred eu, nici viteza, nici capacitatea mare a memoriei operative sau alte performanțe hard, nici chiar limbajul de programare, oricît de exotice sau natural ar fi, ci, mai curînd, mutația pe care o vor impune, și o vor susține prin toate acestea, în modul de a gândi programarea, mai apropiat de modul natural de rezolvare a problemelor. Utilizatorilor li se cere, deci, o conștientizare a procesului natural de rezolvare, acesta fiind descris în limbaj natural sau pseudonatural, pentru a deveni program.

În consecință, nivelul de detaliere a algoritmului de rezolvare va fi mai scăzut decît cel obișnuit programării clasice. Centrul de greutate al specificării procesului de rezolvare se va deplasa din domeniul structurării rigide a transformărilor datelor spre domeniul precizării evenimentelor, obiectelor și relațiilor dintre acestea, caracteristice problemei.

Precizarea la nivel general a modului de rezolvare a problemelor, specificarea lor în limbaj natural, utilizarea unor structuri interne speciale și unor metode foarte generale de prelucrare a specificațiilor de nivel înalt, cad sub incidența inteligenței artificiale (IA). Este natural, deci, ca noua generație de calculatoare să se bazeze, printre altele, pe rezultatele cercetărilor din acest domeniu.



Figura 1 ilustrează schematic bariera ce va fi trecută și care, de fapt, constă în schimbarea sensului de a privi programarea. În loc să ne situăm în exteriorul 'zonei inteligente' și să respectăm constrîngerile unor structuri impuse construcției algoritmilor, unor limbaje de programare și

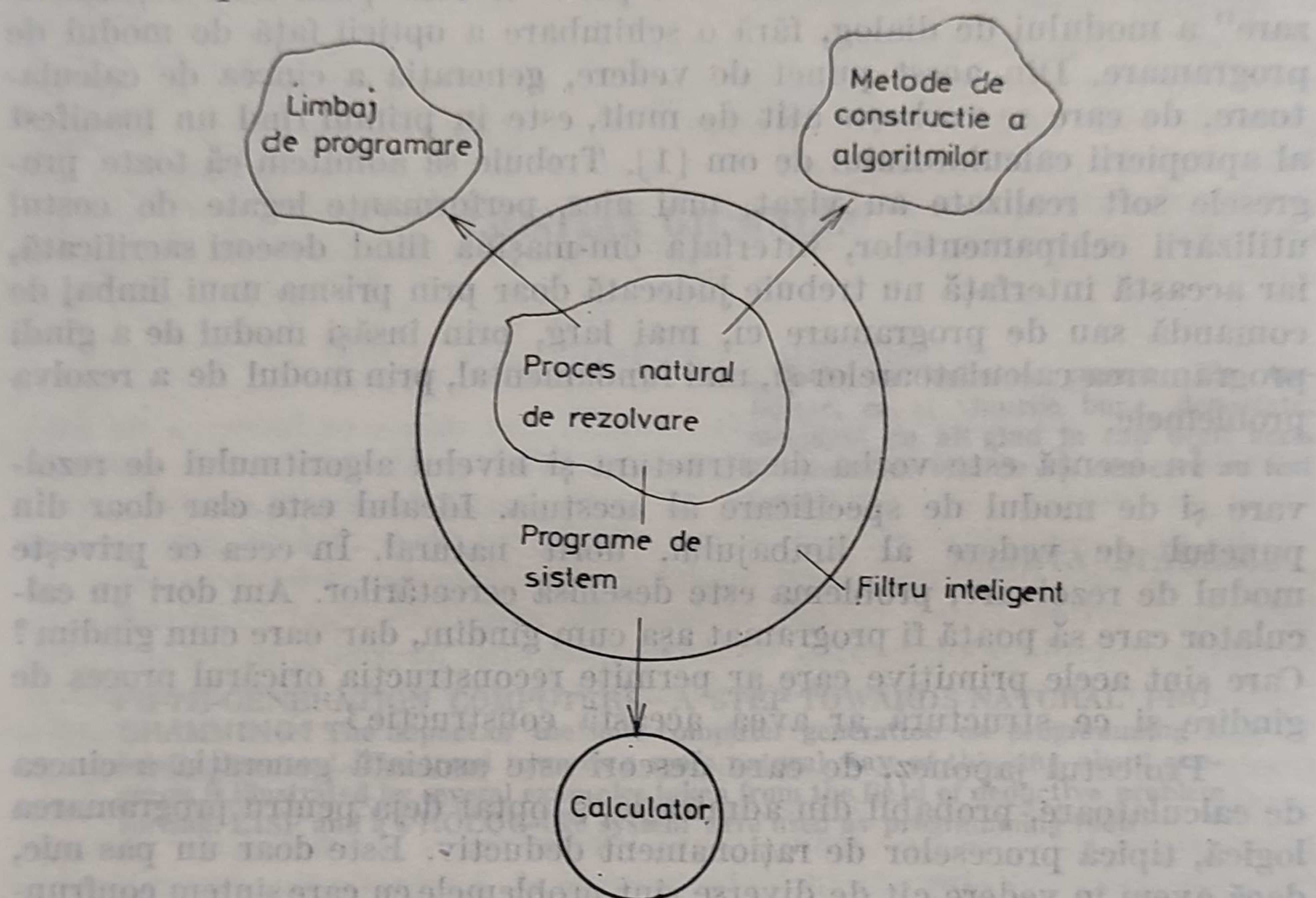


Fig. 1. — Programare „naturală”.

unor mașini de calcul date, extrapolînd inteligența, vom rămîne în interiorul zonei inteligente filtrînd specificațiile vagi de rezolvare prin lentila programelor de sistem cu care vor fi dotate calculatoarele noii generații.

### 3. Moduri de rezolvare a problemelor

Pentru a concretiza, într-o oarecare măsură, cele de mai sus, și pentru a remarca sarcinile formidabile ce revin programelor de sistem ale noilor generații de calculatoare, să privim puțin, pe baza cîtorva exemple foarte simple, procesul de rezolvare a unor tipuri particulare de probleme, așa cum îl realizăm noi sau, cel puțin, așa cum ni se pare că îl realizăm.

În primul rînd, să observăm că există tipuri foarte diferite de probleme ce impun moduri diferite de raționament. Este posibil să fie imaginate metode foarte generale de rezolvare [5], dar, în schimb, extrem de ineficiente. Cercetările din IA în domeniul rezolvitoarelor generale de probleme au demonstrat acest lucru. Ca alternativă, fundamentarea unor metode mai eficiente, dar aplicabile doar anumitor clase de probleme, pare mai atrăgătoare și justificată tehnologic. Chiar și așa, există extrem de multe căi deschise cercetării.



În exemplele următoare va trebui să facem distincție între două niveluri de rezolvare, cel specificat de utilizator și cel intern, implantat pe calculator, practic invizibil utilizatorului. Vom observa că o mare parte din algoritmul de detaliu propriu rezolvării este preluat de nivelul intern, nivelul extern concentrând doar informații cu caracter mai mult declarativ. În fond, utilizatorul va specifica 'ce trebuie făcut', 'cum trebuie făcut' rămânând exclusiv în sarcina nivelului intern. De aici derivă nevoia unor metode generale de rezolvare,

Programele aferente problemelor rezolvate sînt scrise în KISS [6], un sistem asemănător PROLOG-ului [2], și pentru comparație, în LISP, limbajul cel mai utilizat, în prezent, în IA.

#### 4. Rezolvare prin deducție „monolit”, programare logică

Acesta este domeniul cel mai fertil la ora actuală, existînd deja aplicații rezolvate de sisteme ce simulează procesul deductiv, controlat prin intermediul unor baze de cunoștințe, sisteme denumite „de tip expert”. Este de așteptat ca noua generație de calculatoare să ofere arhitecturi hard adecvate unor astfel de programe. Limbajul reprezentativ, folosit pentru programarea deductivă, este PROLOG-ul, gîndit ca limbaj mașină pentru noua generație, în viziunea proiectului japonez, întregul eșafodaj soft de nivel înalt urmînd a fi construit în PROLOG.

Să considerăm următoarea problemă [3], rezolvabilă printr-un proces deductiv simplu :

##### *Problema 1*

Se cere proiectarea unui apartament format din două camere respectînd următoarele condiții :

- 1a) Camerele sînt rectangulare și orientate nord-sud sau est-vest.
- 1b) Fiecare cameră are o singură fereastră și o singură ușă interioară, cu excepția uneia dintre camere echipată, în plus, cu o ușă exterioară.
- 1c) Camerele comunică prin ușile interioare.
- 1d) Ferestrele nu pot fi orientate spre nord.
- 1e) Ferestrele trebuie plasate pe pereții exteriori.

Rezolvarea acestei probleme, prin Programul 1, traversează trei etape :

— Stabilirea limbajului folosit pentru specificarea problemei (secțiunea SABLOANE).

— Specificarea propriu-zisă a problemei, în particular a obiectelor problemei, a relațiilor considerate întotdeauna adevărate (eg. nord opus cu sud) și a relațiilor de satisfăcut în vederea rezolvării propriu-zise (secțiunea REGULI).

— Precizarea unor funcții elementare, asociate unora dintre șabloane. Considerînd șabloanele ca functori distribuiți, procesul de rezolvare decurge la fel ca și în PROLOG, legea de deducție fiind *modus-ponens*, iar strategia de aplicare *backtrack*. În realitate, programul structurează,



sub formă de arbore, așa cum se arată în figura 3, spațiul stărilor posibile corespunzătoare soluțiilor, arbore parcurs în adâncime, în vederea determinării unei soluții.

Programul 1 reprezintă nivelul superficial al specificării modului de rezolvare. Un element important al acestui proces, anume mașinăria de control al procesului deductiv, mai precis mașinăria backtrack, este total ascunsă programatorului, fiind integrată calculatorului specia-

CONTEXT PLAN (> \_)

SABLOANE

```
(plan
  (proiect_cămeră_intrare _ _ _).
  (proiect_camera _ _).
  (alege _).
  (_ opus cu _).
  (_ este_diferit_fata_de _ ) dif.
```

REGULI

```
(plan _UE _U1 _F1 _U2 _F2)
  (proiect_cămeră_intrare _UE _U1 _F1)
  ( _U1 opus cu _U2)
  (proiect_camera _U2 _F2).
  (proiect_camera_intrare _UE _U _F)
  (proiect_camera _U _F)
  (alege _UE)
  ( _UE este_diferit_fata_de _U).
  (proiect_camera _U _F)
  (alege _U) (alege _F)
  ( _U este_diferit_fata_de _F)
  ( _F este_diferit_fata_de_nord).
  (nord opus cu sud).
  (sud opus cu nord).
  (est opus cu vest).
  (vest opus cu est).
  (alege sud).
  (alege nord).
  (alege est).
  (alege vest).
```

FUNCTII

```
(DEFUN dif (x y)
  (NOT (EQUAL x y))
)
```

STOP

```
?? (plan _UE _U1 sud _U2 _F2).
= - (plan sud nord sud sud est)
??
= - (plan sud nord sud sud vest)
??
= - (plan est nord sud sud est)
```

Programul 1

lizat capabil să execute cod KISS. De aici ideea de a construi mașini specializate în care operațiile elementare nu mai sînt binecunoscutele operații aritmetice, logice și de salt ale calculatoarelor convenționale, ci, în plus, operații de cu totul altă natură și mult mai complexe, cum ar fi în cazul de față unificarea logică, căutarea simbolurilor într-o tabelă de dispersie, suprapunerea unui șablon peste un șir de simboluri, primitive pentru construcția și parcurgerea arborilor etc. Mașinile de acest tip sînt, uneori, denumite non-von Neumann, deși, deocamdată, ele sînt simulate din plin pe calculatoare cu structură clasică.



```

(DEFUN test_final (sol)
  (= (LENGTH sol) 5)
)
(DEFUN pot_sa_extind? (sol alt)
  (COND ((NULL sol) )
        ((= (LENGTH sol) 1)
         (NOT (EQ (CAR sol) alt) )
        )
        ((= (LENGTH sol) 2)
         (AND (NOT (EQ (CADR sol) alt))
               (NOT (EQ alt 'nord))
         )
        )
        ((= (LENGTH sol) 4)
         (AND (NOT (EQ (CADDR sol) alt))
               (NOT (EQ alt 'nord))
         )
        )
        ((OR (AND (EQ (CADR sol) 'sud)
                   (EQ alt 'nord)
                 )
              (AND (EQ (CADR sol) 'nord)
                    (EQ alt 'sud)
                 )
              (AND (EQ (CADR sol) 'est)
                    (EQ alt 'vest)
                 )
              (AND (EQ (CADR sol) 'vest)
                    (EQ alt 'est)
                 )
            )
        )
  )
)
(DEFUN tiparire_plan (sol)
  (MAPC (LAMBDA (nume val)
        (PRINC nume)
        (PRINC " = ")
        (PRINT val)
      )
        (UE U1 F1 U2 F2)
        (REVERSE sol)
  )
)
(DEFUN EXTIND (sol alternative)
  (COND ((test_final sol)
        (tiparire_plan sol)
      )
        ((pot_sa_extind? (REVERSE sol) (CAR alternative) )
        (COND ((EXTIND
                (CONS (CAR alternative) sol)
                ('nord est sud vest)
              )
              ((EXTIND sol (CDR alternative) )
              )
            )
        )
  )
)

```

#### Programul 2



Oiștigul modului de programare ilustrat, cunoscut sub numele de programare logică, deoarece regulile nu sînt altceva decît propoziții Horn, este evident atît din punctul de vedere al conciziunii cît și al

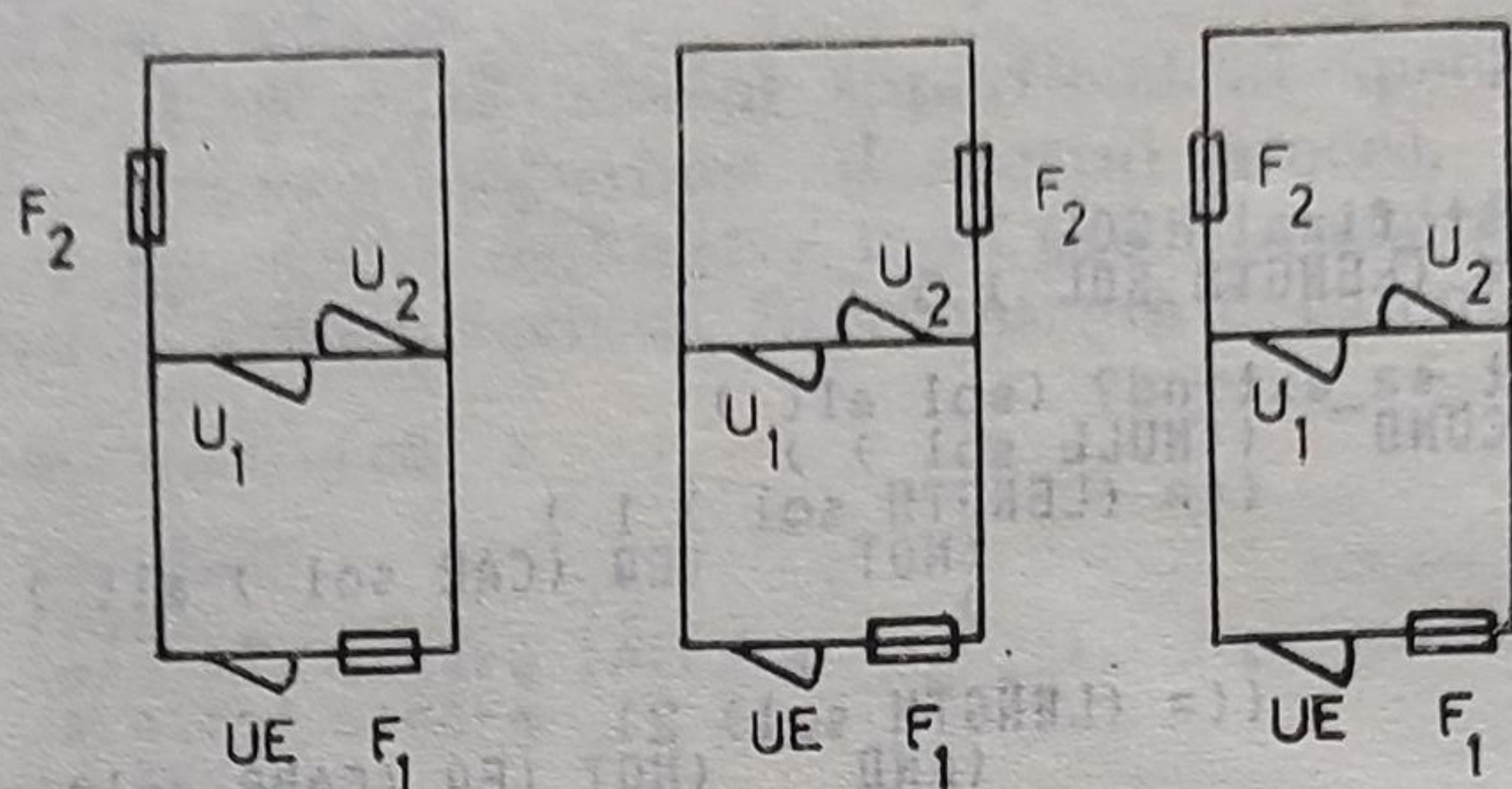


Fig. 2. — Planuri obținute prin Programul 1.

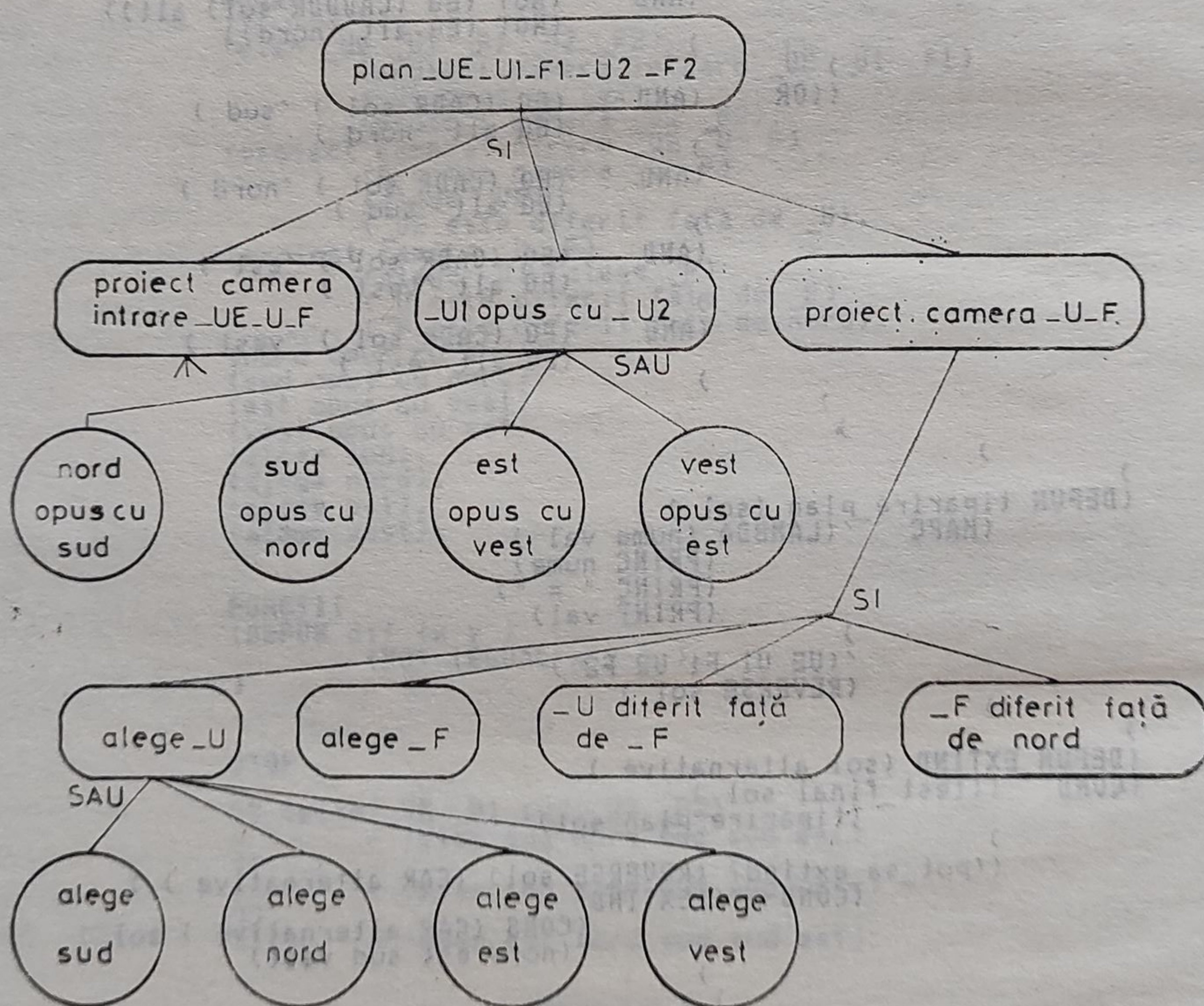


Fig. 3. — Un fragment al arborelui SI-SAU parcurs de Programul 1.

altor factori mai importanți, cum ar fi claritatea și posibilitatea de a modifica foarte ușor programul. Pentru a reliefa acest oiștig, Programul 1 a fost rescris în LISP (vezi Programul 2), punîndu-se în evidență mecanismul backtrack, independent de problemă. Într-adevăr, funcția EXTIND realizează parcurgerea în adîncime a arborelui ilustrat în figura 3, folosind funcțiile ajutătoare, dependente de problemă, pot\_să\_extind și test\_final, precum și mulțimea alternativelor posibile de extindere a soluțiilor



parțiale. EXTIND pornește cu soluția vidă și inserează, rînd pe rînd, la începutul listei sol, pozițiile corespunzătoare elementelor UE, U1, F1, U2, F2, toate acestea avînd aceeași semnificație ca și în Programul 1. Inserarea oricărui element în soluție este validată prin funcția pot\_să\_extind, care 'își dă seama' de natura elementului inserat în funcție de lungimea listei sol (eg. (LENGTH sol) = 2 implică (CAR sol) = poziție U1 și alt = poziție F1).

Programul LISP este mult mai puțin clar decît echivalentul său KISS, deși LISP-ul este recunoscut prin conciziunea și eleganța sa. În plus, programul nu poate funcționa cu soluții parțiale, propuse de utilizator, așa cum se întîmplă cu Programul 1. Mai mult, unui nespecialist programul LISP îi este inaccesibil, în timp ce specificația declarativă din programul 1 poate fi acceptată relativ ușor. Din nefericire nu toate problemele se pretează unei rezolvări pur deductive și, să recunoaștem, eficiența rezolvării problemelor prin orice lege de inferență controlată prin backtracking este extrem de scăzută. Aici viteza este sacrificată generalității.

## 5. Rezolvare prin cooperarea unor procese deductive

Un pas semnificativ către un mod mai natural de raționament, deductiv eventual, este făcut prin fragmentarea procesului de rezolvare, considerat pînă acum monolit, în părți, numite procese, în cazul de față deductive, capabile să lucreze independent. Iată o problemă, foarte cunoscută în IA, care se pretează bine acestui mod de rezolvare.

### Problema 2

Se consideră realizarea unei construcții din cuburi, pe baza unui model dat, așa cum se exemplifică în figura 4. Spațiul de lucru, numit CONSTR, conține inițial o mulțime de cuburi așezate unele peste altele sau pe 'masă' într-un mod oarecare. Cuburile trebuie deplasate astfel încît, în final, configurația din CONSTR să fie cea corespunzătoare modelului. Deplasările cuburilor sînt supuse următoarelor restricții:

- 2a) Un cub poate susține în mod direct un singur cub.
- 2b) Suportul numit 'masă' poate susține oricîte cuburi.
- 2c) Pentru a putea fi deplasat un cub trebuie să fie liber, adică să nu susțină, direct sau tranzitiv, alte cuburi.

Cum ne-am imagina rezolvarea acestei probleme? Probabil că, privind modelul și construcția curentă, ne-am propune multe mutări, unele imediat realizabile, altele depinzînd de mutări prealabile. De exemplu, mutarea (b pe c) nu poate fi realizată cît timp b nu este liber, iar ca b să fie liber trebuie să nu susțină nici pe a — deși (a pe b) face parte din starea finală — nici pe c. Prin urmare, unele mutări pierd din importanță la un moment dat, altele devenind mai ușor de realizat, ordonarea mutărilor de efectuat avînd loc dinamic, în funcție de starea construcției. De asemenea, unele mutări realizate nu mai captează atenția. Astfel, după ce b este deja pe c, iar c este pe masă, și dacă procedăm inteligent, mutarea (b pe c) nu va mai fi niciodată repetată.



Într-o terminologie mai rigidă vom spune că, la un moment dat, în mintea rezolvitorului coexistă mai multe scopuri a căror satisfacere concură la rezolvarea problemei. Fiecare scop este rezolvat de către un proces de rezolvare independent, iar dacă rezolvarea se face deductiv, la fel ca în Programul 1 procesul este 'de inferență logică'. În cazul de față, fiecare relație dintre două blocuri, în model, reprezintă un scop. Prin urmare, pentru modelul din figura 4 vor exista inițial trei procese de inferență, simbolizate  $P_{(a \text{ pe } b)}$ ,  $P_{(b \text{ pe } c)}$  și  $P_{(c \text{ pe masă})}$ . Fiecare proces are, în cursul rezolvării, un 'merit' care, într-un fel, reprezintă interesul rezolvitorului față de procesul în cauză. Meritul proceselor variază odată cu modificarea construcției, deci încercarea de a rezolva unele procese poate antrena modificarea meritului proceselor. Orice proces poate fi creat, activat, suspendat, reluat și distrus, în mod controlat. Modul de control este cu adevărat interesant deoarece, spre deosebire de cazurile obișnuite ale sistemelor de operare, nu este înghețat în vreun algoritm de control exterior proceselor, ci este un control 'oportun', localizat în fiecare proces, și sensibil la modificările universului problemei rezolvate.

De exemplu, dacă inițial ordinea proceselor, funcție de merit, este :

$$P_{(b \text{ pe } c)} > P_{(c \text{ pe masă})} > P_{(a \text{ pe } b)},$$

prin însăși starea construcției se va realiza scăderea meritului procesului  $P_{(b \text{ pe } c)}$  în așa fel încât acesta să fie mai puțin important ca  $P_{(c \text{ pe masă})}$ , secvența de monotonii ale proceselor fiind :

$$P_{(c \text{ pe masă})} > P_{(b \text{ pe } c)} > P_{(a \text{ pe } b)} \text{ implică } (c \text{ pe masă}).$$

$$P_{(b \text{ pe } c)} > P_{(a \text{ pe } b)} \text{ implică (liber } b) \text{ care implică } (a \text{ pe masă}) \text{ și } (b \text{ pe } c).$$

$$P_{(a \text{ pe } b)} \text{ implică } (a \text{ pe } b).$$

Programul 3 descrie destul de clar modul de raționament și, de fapt, controlul execuției proceselor. Toate procesele de mutare a cuburilor lucrează în contextul\*) numit LUCRU și au, inițial, același merit.

Un proces  $P_{(-x \text{ pe } -y)}$ , unde  $-x$  și  $-y$  desemnează obiecte generice, fiind variabile, se comportă astfel :

a) Dacă există un proces  $P'$  al cărui scop satisface șablonul  $(-y \text{ pe } -)$ , deci  $P'$  încearcă să mute  $-y$ , atunci, evident,  $P$  va trebui să aștepte satisfacerea lui  $P'$ . Logica de realizare a așteptării este concentrată în contextul ASTEPTARE.

b) Dacă nu există nici un proces  $P'$  care vizează deplasarea lui  $-y$ , iar  $(-x \text{ pe } -y)$  există deja în spațiul construcției, atunci  $P$  este distrus.

c) Dacă nu există nici un proces  $P'$ , care încearcă mutarea lui  $-y$ , dar  $(-x \text{ pe } -y)$  nu există în spațiul construcției, atunci scopul procesului  $P$  va fi satisfăcut imediat. Se eliberează  $-x$ , apoi  $-y$ , urmînd, în final, mutarea lui  $-x \text{ pe } -y$ . Procesul  $P$  va fi acum distrus.

\*) În KISS, contextele sînt asemănătoare tablelor de lucru din sistemele de reprezentare și prelucrare a cunoștințelor concentrînd sursele de cunoștințe (alci reguli) necesare rezolvării unor tipuri de subprobleme.



CO INIT.

```
##
Acest context grupeaza informatiile necesare formarii
proceselor de rezolvare caracteristice unei probleme
particulare in lumea cuburilor. Problema este prezentata
in forma unui scop multiplu (a pe b) (c pe masa) etc.
Pentru fiecare subscop de tipul (x pe y) se genereaza
un proces de rezolvare cu subscopul respectiv.
##
```

SABLOANE

```
( _ pe _ ).
(creeaza _ merit _ in _ cu scopul + ) #CREATE.
(reluare _ ) #RESUME.
(distrugere _ ) #KILL.
( _ ).
```

REGULI

```
( _X pe _Y )
(creeaza GLOBAL _proces merit 50. in LUCRU cu scopul
( _X pe _Y ) ).
(executa )
(creeaza GLOBAL START merit 0 in HERE cu scopul
(reluare BEST )
(distrugere THIS ) ).
```

CO LUCRU.

```
##
Contextul LUCRU contine principalele reguli
de satisfacere a scopului asociat unui proces
particular de rezolvare.
```

##

SABLOANE

```
( _ pe _ ).
(elibereaza _ ).
(muta _ pe _ ).
(asteapta satisfacerea scopurilor + ) ASTEPTARE.
(+ exista in CONSTR ? ) CONSTR.
(sterge + din CONSTR ) CONSTR.
(inregistreaza + in CONSTR ) CONSTR.
(distrugere _ ) #KILL.
(reluare _ ) #RESUME.
(tiparire + ) tiparire.
( _ ).
```

REGULI

```
( _x pe _y )
(asteapta satisfacerea scopurilor ( _y pe _ ) )
#
( _x pe _y exista in CONSTR ? )
(distrugere THIS )
(reluare BEST ).
( _x pe _y )
(elibereaza _x )
(elibereaza _y )
(muta _x pe _y )
(distrugere THIS )
(reluare BEST )
(elibereaza masa ).
(elibereaza _x )
( _z pe _x exista in CONSTR ? )
(elibereaza _z )
(muta _z pe masa ).
(elibereaza _x ).
(muta _x pe _y )
( _x pe _z exista in CONSTR ? )
(sterge _x pe _z din CONSTR )
(inregistreaza _x pe _y in CONSTR )
(tiparire muta _x pe _y ).
```



# CO AȘTEPTARE.

```
##
Orice proces cu un scop de tipul ( X pe _cub ) este
menținut în așteptare pînă ce _cub va fi poziționat ca în
model.
##
```

```
SABLOANE
(așteapta satisfacerea scopurilor + ).
(merit = 0) #RESOLUT.
(reluare = 0) #RESOLUT.
(exista un proces _ cu scopul _ + ? ) #SELPROC.
( _ ).
```

```
REGULI
(așteapta satisfacerea scopurilor _scop )
(exista un proces _proces cu scopul OR _scop ? )
(merit _proces = mp )
(merit THIS = mp )
(merit THIS = 1 )
(reluare BEST )
(așteapta satisfacerea scopurilor _scop ).
```

# CO CONSTR.

```
##
Acest context conține starea curentă a lumii cuburilor
Bineînțeles, la începutul rularii programului, CONSTR
specifică starea inițială.
##
```

```
SABLOANE
( _ pe _ ).
( _ pe _ exista în CONSTR ? ).
( _ pe _ exista în CONSTR ) #GENP.
(sterge + from CONSTR ) #REMP.
```

```
REGULI
(c pe a ).
(a pe b ).
(b pe masa ).
( _x pe _y exista în CONSTR ? - ).
( _x pe _y ).
```

# CO INIT.

```
##
Rezolvarea propriu-zisă a unei probleme particulare
în lumea cuburilor se realizează prin formularea unui
scop ca, de exemplu
##
```

```
?
(a pe b )
(b pe c )
(c pe masa ).
```

```
LP. ## Sint create următoarele procese ##
```

# PROCESE

```
- P00006 MERIT= 50. CONTEXT= LUCRU
PROPRIETATI
SCOPURI
- (c pe masa )
- P00005 MERIT= 50. CONTEXT= LUCRU
PROPRIETATI
SCOPURI
- (b pe c )
- P00004 MERIT= 50. CONTEXT= LUCRU
PROPRIETATI
SCOPURI
- (a pe b )
```

```
? (executa).
(muta c pe masa )
(muta a pe masa )
(muta b pe c )
- (muta a pe b )
```



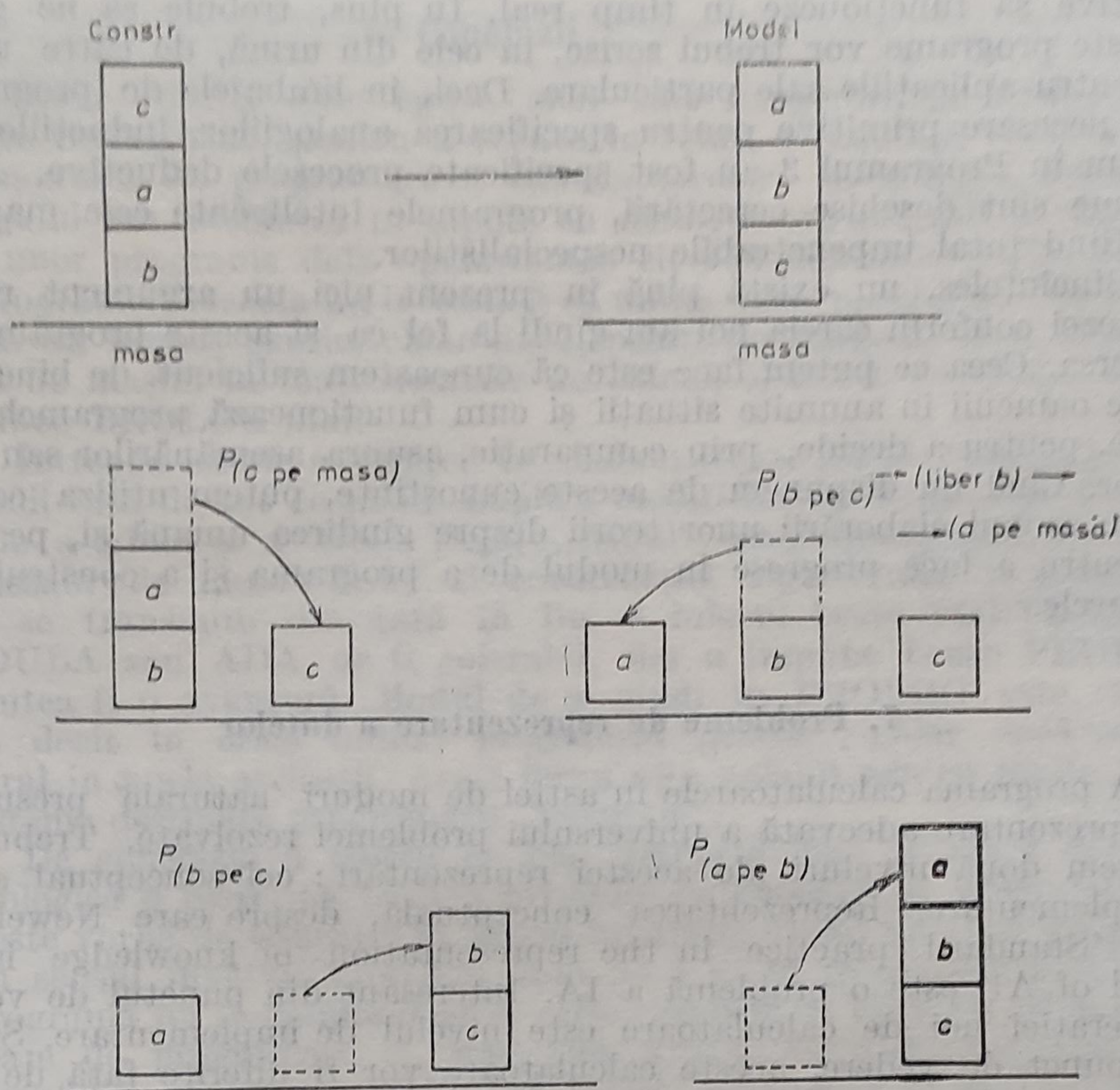


Fig. 4. — O construcție în lumea cuburilor.

Bineînțeles, strategia de creare și control a proceselor descrisă aici este doar una din numeroasele alternative posibile, fenomenul esențial fiind reprezentat de fragmentarea problemei în subprobleme și rezolvarea în paralel a acestora.

Procesele de inferență reprezintă un instrument puternic pentru simularea unor moduri naturale de raționament. În exemplul dat, procesele sînt pur deductive, dar, în esență, ne putem imagina procese inductive sau care rezolvă scopuri prin analogie. În orice caz, soluția unei probleme dificile nu se conturează clar din primele momente ale rezolvării, unele căi de rezolvare, care la început par promițătoare, pot fi abandonate sau ulterior reluate în funcție de mersul rezolvării și de rezolvitor. Această dinamică poate fi simulată bine prin intermediul proceselor de raționament asociate unor subscopuri.

## 6. Alte moduri de rezolvare

Rezolvarea deductivă este aplicabilă doar unei clase limitate de probleme. Dacă dorim ca mașina noilor generații să devină cu adevărat un partener inteligent al utilizatorilor, atunci programele cu care este echipată vor trebui să realizeze și analogii, inducții și, în ultimă instanță, să învețe. Există deja încercări și realizări în aceste direcții, nedepășind însă faza de laborator [4], cu atît mai mult cu cît dorim ca programele



respective să funcționeze în timp real. În plus, trebuie să ne gândim că aceste programe vor trebui scrise, în cele din urmă, de către utilizator, pentru aplicațiile sale particulare. Deci, în limbajele de programare vor fi necesare primitive pentru specificarea analogiilor, inducțiilor etc. așa cum în Programul 3 au fost specificate procesele deductive. Aceste probleme sînt deschise cercetării, programele inteligente cele mai evoluat fiind total impenetrabile nespecialiștilor.

Bineînțeles, nu există pînă în prezent nici un argument riguros al ipotezei conform căreia noi am gândi la fel ca și aceste programe sau vice-versa. Ceea ce putem face este că cunoaștem suficient de bine cum gîndesc oamenii în anumite situații și cum funcționează programele respective, pentru a decide, prin comparație, asupra asemănărilor sau diferențelor. Cînd nu dispunem de aceste cunoștințe, putem utiliza comparația în scopul elaborării unor teorii despre gîndirea umană și, pe baza lor, pentru a face progrese în modul de a programa și a construi calculatoarele.

## 7. Probleme de reprezentare a datelor

A programa calculatoarele în astfel de moduri 'naturale' presupune și o reprezentare adecvată a universului problemei rezolvate. Trebuie să distingem două niveluri ale acestei reprezentări: cel conceptual și cel de implementare. Reprezentarea conceptuală, despre care Newell [7] spune 'Standard practice in the representation of knowledge is the scandal of AI' este o problemă a IA. Interesant din punctul de vedere al generației noi de calculatoare este nivelul de implementare. Și din acest punct de vedere, aceste calculatoare vor fi diferite față de cele contemporane.

Reprezentarea datelor implică nu numai anumite primitive de structurare și prelucrare, ci, în același timp, impune un anumit mod de referire și acces la date. Chiar intuitiv, se observă că în programele exemplificate aici nu se mai lucrează cu locații de memorie și adrese ci, mai curînd, cu simboluri și structuri formate cu simboluri. Identificarea nu se mai face prin adresă ci prin conținut, fiind, deci, asociativă.

Dacă în prezent structura conceptuală a reprezentărilor, de exemplu rețele semantice, frame-uri, reguli, este implementată folosind structuri de date și tehnici adecvate calculatoarelor 'clasice', probabil că în viitor acest ecart se va micșora. Calculatorul proiectului japonez va fi, se speră, o mașină PROLOG, capabilă să reprezinte și să prelucereze reguli. De asemenea, există deja mașini LISP, capabile să lucreze în mod direct cu structuri de tip listă. Oricum, însă, a apărut problema precizării unor structuri elementare de reprezentare, altele decît biencunoscuta celulă de memorie adresabilă în mod direct, suficient de flexibilă pentru a permite implementarea structurilor simbolice necesare noilor generații de programe. Este o problemă fundamentală care suscită deja o competiție între 'termenul PROLOG' și 'CONS-ul LISP'.

Bineînțeles, introducerea unor noi metode de reprezentare primară a datelor va impune și noi primitive de prelucrare și control. Dacă în unele cazuri acestea sînt clare, cum este în LISP, nu aceeași este situația altor reprezentări. Chiar și proiectul japonez admite că prima mașină PROLOG va fi un mutant cu multe din caracteristicile arhitecturii 'clasice'.



## 8. Concluzii

Unele, dintre cele spuse aici, sînt exagerări, pe termen scurt, folosite intenționat, pentru a scoate în relief o tendință certă a evoluției modului de programare a calculatoarelor: micșorarea diferențelor dialogului om-calculator în raport cu dialogul om-om, atît în sfera utilizării unor programe deja operaționale cît și în domeniul elaborării unor noi programe. Aceasta nu înseamnă că mâine vom programa în PROLOG sau în alt limbaj exotic, chiar dacă, printr-un miracol, ar fi disponibile astfel de mașini în toate centrele de calcul, și oricît de multă reclamă s-ar face PROLOG-ului.

Întreaga istorie a științei ne demonstrează cît de conservatori sînt oamenii față de tot ce poate implica modificări fundamentale în modul lor de a gîndi și a trăi. Deja, „tribul” programatorilor și-a format un tezaur de „know-how” al construcției algoritmilor și programelor care se transmite din tată în fiu. A înlocui brusc FORTRAN-ul cu MODULA sau ADA, ar fi tolerabil, dar a impune brusc PROLOG-ul ar putea fi o aventură. Modul de a gîndi în PROLOG este cu totul altul decît în orice limbaj procedural „clasic”, chiar dacă este mai natural în unele aplicații. Acest lucru este valabil pentru multe limbaje și sisteme destinate programării „naturale”.

De generația a cincea ne mai desparte încă, în afară de bariera tehnologică și teoretică, bariera nivelului de educație al programatorilor. Nu este vorba de educație în sensul învățării unui limbaj de programare ci în sensul învățării unui nou mod de a gîndi rezolvarea problemelor. A programa nu înseamnă doar a scrie excelent în FORTRAN sau în alte limbaje de programare ci, mai ales, a avea forța de conștientizare a procesului de rezolvare a aplicației, astfel încît acesta să poată fi adaptat primitivelor de reprezentare și control proprii oricărui tip de limbaj de programare. Din acest punct de vedere, programarea „naturală” este mai pretențioasă decît programarea „clasică”. Deja, în domeniul ingineriei cunoașterii, domeniu al IA, se admite că problema cea mai dificilă este cea a achiziției cunoștințelor, a descoperirii modului în care un expert într-un domeniu gîndește rezolvarea problemelor tipice domeniului.

Generațiile viitoare de calculatoare vor cere și o nouă generație de programatori, iar pentru cei care cred că mașinile de calcul ale viitorului vor fi programate „cu vîrfurile minții”, amintesc cîteva cuvinte ale lui Nichita Stănescu:

„Cui i-a dat vreodată prin cap și cine și-ar putea imagina vreodată că prin crearea unui ceas perfecționat se poate crea și timp?”.

## BIBLIOGRAFIE

1. \* \* \* New Computing Generation, 1, July (1983).
2. W. F., CLOCKSIN, C. S. MELLISH, *Programming in PROLOG*, Springer, 1981.
3. \* \* \* *How to solve it with PROLOG*, Laboratório Nacional de Engenharia Civil, Lisboa, 1978.
4. A. BARR., E. A. FEIGENBAUM, (Ed.), *The handbook of artificial intelligence*, Vol. 111, Stanford Press, 1982.
5. A. NEWELL, *Artificial intelligence and the concept of mind*, in *Computer models of thought and language*, Freeman Co., 1973.
6. C. A. GIUMALE, *A Knowledge Inference Simple System*, manual de referință, Cat. Calculatoare, IPB, 1983.
7. A. NEWELL, *The knowledge level*, Artificial intelligence, 18, 1, January (1982).



# DIALISP — EXPERIMENT DE STRUCTURARE NECONVENȚIONALĂ A UNEI MAȘINI LISP

AUREL PĂUN\*) GHEORGHE M. ȘTEFAN\*),  
ANDY BIRNBAUM\*), VIRGIL BISTRICEANU\*)

DIALISP — AN EXPERIMENT WITH A NONCONVENTIONAL LISP MACHINE. The design of a Lisp Machine requires the intensive use of parallel processes assumed by language implementation strategies, as well as a deeper discount of its main functions into the physical structure. Another specific feature of LISP is the need of a large amount of main memory. The paper presents a family of LISP machines.

## 1. Introducere

Criza actuală în domeniul tehnicii de calcul a debutat prin aceea că realizarea și testarea programelor de mare complexitate a devenit deosebit de greoaie și de imprecisă. Acest fenomen se petrece în contextul în care actualizarea funcțională a structurilor digitale se face din ce în ce mai mult prin intermediul tehnicilor de programare, microprogramare sau nanoprogramare. Creșterea preciziei și complexității funcționale se face prin ridicarea exagerată a prețului de cost asociat programării unor structuri ce în mod constant se realizează la prețuri de cost din ce în ce mai reduse. Această situație, la limită paradoxală, a impus stagnarea acestui mod de evoluție ce presupunea optimizarea exagerată a hardware-ului numai din propria sa perspectivă cu implicații negative asupra eficienței în punerea la punct a software-ului.

Structurile digitale posedă încă dubla dimensiune dată de structura fizică și cea informațională, iar optimizarea desfășurată numai pe una dintre dimensiuni, de multe ori pe seama celeilalte, nu poate decât să genereze un ansamblu deficitar. Raportul cost-performanță tinde să crească în acest caz supărător. Optimizarea structurii fizice pînă la limita în care prețul ei de cost, în ansamblul mașinii digitale, devine neglijabil, creează toate condițiile ca un parametru, structura fizică, să nu mai poată fi manipulat pentru a se reduce prețul de cost. Dimensiunile pe care poate acționa proiectantul arhitecturii de sistem reducîndu-se cu una, dar la una singură, posibilitățile de optimizare tind să fie anulate.

Limbajele și stilul de programare au fost cele ce au permis și stimulat această evoluție. La aceasta s-au adăugat restricții tehnologice ce inițial au orientat configurarea sistemelor de calcul.

În momentul de față corelația dintre structura fizică a sistemelor de calcul și funcțiunile asociate acestora tinde să fie nulă. Procesul de anulare a acestei corelații a fost condiționat de „mijlocirea perfectă” asigurată prin tehnicile de programare. Programele au izolat funcția de structură atît de perfect încît structurile au putut deveni *universale* în condițiile în care funcțiunile s-au *diversificat la maximum*.

\*) Institutul politehnic din București.



Evoluțiile în domeniul structurilor fizice au avut numai efecte cantitative, raportate, de regulă, exclusiv la ele însele. Calculatoarele, ca structură fizică, au evoluat *numai* în sensul în care au devenit din ce în ce mai *putenice* sau mai *compact realizate*.

Un prim pas în sensul rezolvării acestei crize a fost făcut prin conceperea unor noi limbaje de programare, chiar prin imaginarea unor noi stiluri de programare (programare funcțională [1, 2]). S-a urmărit prin aceasta o abordare mai eficientă a problemelor legate de conceperea, elaborarea, testarea și întreținerea programelor. De asemenea, s-au întrevăzut posibilități de realizare mai comodă a anumitor clase de aplicații prin intermediul unor limbaje convenabile. Teoretic, o parte din aceste scopuri au fost atinse, sau se întrevăd soluții viabile pe această cale. Practic, acest prim pas a impus ca necesar un al doilea, datorită faptului că, o primă consecință, poate cea mai importantă, a acestei tentative, a fost aceea că, pe mașinile universale, noile limbaje sau stiluri de programare se dovedesc ineficiente (LISP, PROLOG) sau imposibil de abordat (programarea funcțională).

Noua gândire în domeniul programării va impune o reformulare a structurilor fizice de bază, ce vor permite structurarea noilor mașini astfel că nu vor mai putea fi de tip universal o bună bucată de vreme. În continuare programele vor mijloci adecvarea funcției la structură, dar într-un context în care legătura dintre limbaj și funcție, pe de o parte, și cea dintre limbaj și structură, pe de altă parte, va implica și o relație mai puternică între funcție și structura fizică, care va începe să rejoyce un rol important în determinarea performanțelor și a prețului de cost.

Două vor fi modalitățile prin care structura va începe să joace un rol important, rol pierdut prin universalizarea configurațiilor. În primul rând, structura va începe să joace un rol notabil în declanșarea *mecanismelor paralele*. Paralelismul va căpăta și o semnificație hardware chiar dacă vor exista nuanțe ce-l vor deosebi de cel software. În al doilea rând, se declanșează un proces de migrare a funcțiilor realizate prin programe către *implementarea lor prin hardware*.

Aceste două tendințe sînt puternic sprijinite de dezvoltarea tehnologiilor VLSI ce permit o complexitate structurală mult mai mare.

În lucrarea de față este prezentată o încercare de implementare a unei mașini ce execută direct funcții în limbajul LISP. Vom arăta că gândirea ce stă la baza acestei mașini se înscrie în tendința sumar comentată anterior.

## 2. Opțiuni

În conformitate cu cele anterior spuse, pentru conceperea unei mașini LISP, trebuiau investigate două probleme: (1) posibilitatea de a declanșa procese paralele în procesul de evaluare; (2) măsura în care unele funcțiuni, ce pe o mașină universală erau implementate prin software, se puteau realiza la un nivel cît mai coborît, în vederea creșterii eficienței.

Referitor la (1) putem spune că actualele versiuni ale limbajului LISP nu presupun posibilitatea declanșării spectaculoase a unor procese paralele asociate funcției evaluare. Dar mecanismele de implementare pot



fi imaginate astfel încât să implice declanșarea unor procese paralele. În acest sens se remarcă faptul că definirea recursivă a unor funcții presupune utilizarea intensivă a mecanismelor de stivă. Apelul la o stivă de mari dimensiuni este un proces ce se poate desfășura în paralel cu controlul mecanismului de evaluare. Controlul și stiva presupun un minimum de paralelism ce va fi implementat cu o structură duală de procesare [6].

În cazul implementării limbajului LISP mecanismele esențiale sînt orientate către utilizarea frecventă a acceselor la memorie astfel încît, într-o structură de tip clasic, procesor-memorie, timpul de execuție apare ca fiind limitat de succesiunea acestora. La limită, se poate imagina o structură de procesor astfel încît timpul asociat evaluării să fie dat strict de suma timpilor presupuși numai la lucrul cu memoria. Pornind de la această observație, se poate utiliza un procesor pentru a urmări, controla, procese ce se desfășoară ca o succesiune coerentă de accese la două memorii distincte. Conceptul de procesor dual apare în acest caz ca fiind natural implicat în cazul unei mașini LISP. În figura 1 este prezentată schema de principiu a unei mașini duale. Timpul este

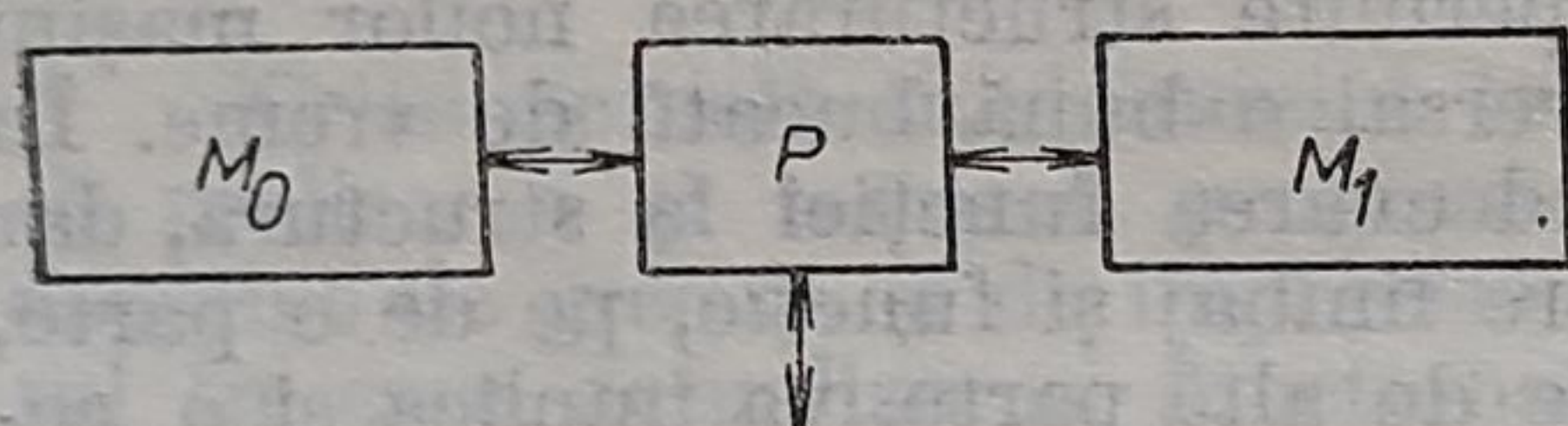
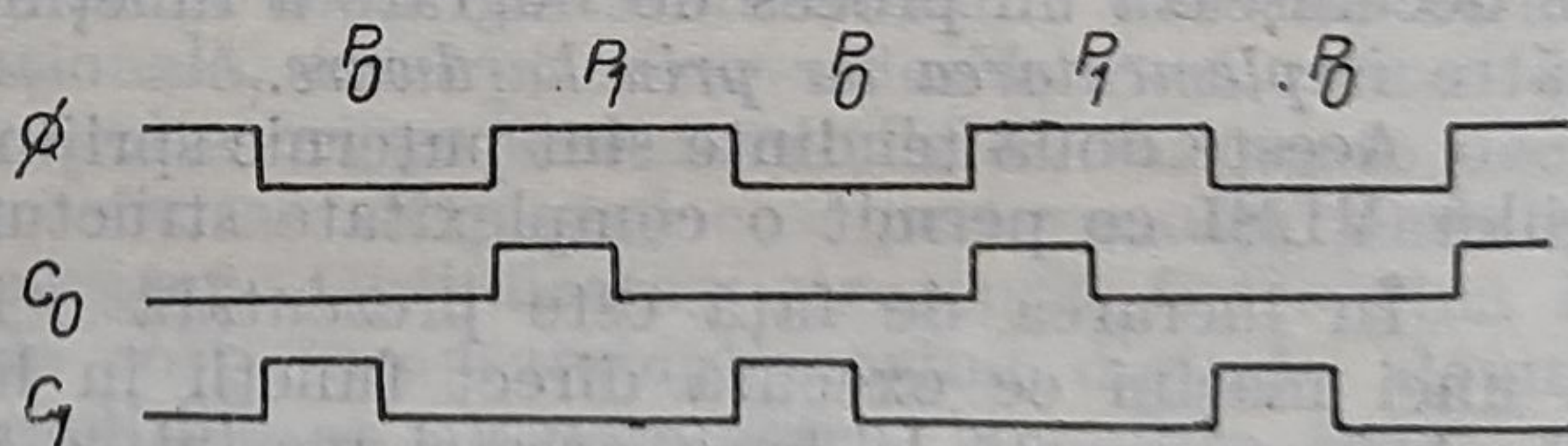


Fig. 1. — Schema de principiu a mașinii duale DIALISP.

distribuit în această structură astfel încît ciclurile de memorie se înlanțuie continuu în  $M_0$  și  $M_1$ , iar procesorul  $P$  își comută controlul asupra datelor din cele două memorii, astfel încît se desfășoară în sistem două procese ce apar ca fiind strict paralele. Formele de undă din figura 2 explicitează modul de lucru al celor trei blocuri prezentate în figura 1.  $C_0$  și  $C_1$ , active pe zero, indică ciclurile de acces la cele două memorii

Fig. 2. —Diagrame de timp pentru procesul dual.



$M_0$  și respectiv  $M_1$ , iar  $\Phi$  marchează intervalele de timp în care procesorul funcționează pentru procesul  $P_0$  ( $\Phi = 0$ ) sau  $P_1$  ( $\Phi = 1$ ). Utilizînd memorii de mare capacitate, realizate în tehnologie MOS, cu cicluri cuprinse între 250 ns și 500 ns se pot proiecta mașini la care perioada semnalului  $\Phi$  să fie cuprinsă între 300 ns și 600 ns. La limită se poate presupune, într-o implementare ce utilizează perfect echilibrat structura, că o funcție ce presupune  $n$  accese la memorie se poate realiza în  $(n/2) \cdot (300 \div 600)$  ns. Pentru  $n = 8$  am obține astfel un timp cuprins între 1,2  $\mu$ s și 2,4  $\mu$ s. Estimarea este făcută în condiții exagerat de optimiste: utilizarea perfect echilibrată a structurii duale. Estimarea cea mai pesimistă ar presupune o dezechilibrare extremă deci un timp egal cu  $n \cdot (300 \div 600)$  ns. În situația anterioară, pentru  $n = 8$  am obține timpi cuprinși între 2,4 și 4,8  $\mu$ s. Ținînd cont de domeniul în care poate varia



ciul unei memorii și de domeniul în care poate varia funcționarea echilibrată a structurii duale se obține, pentru o funcție ce necesită opt accese la memorie, un timp de execuție cuprins între 1,2 și 4,8  $\mu$ s. Riscînd o mediere, am putea spune că o funcție LISP cu opt accese la memorie se realizează în aproximativ 3  $\mu$ s, deci obținem 375 ns/acces la memorie în procesul de evaluare.

Aparent, câștigul nu este notabil dar cifrele nu sînt capabile să pună în evidență în întregime efectele paralelismului într-o structură hardware, datorită faptului că nu se poate estima efectul „decuplării” funcțiilor de control. Procesul  $P_0$  fiind dedicat, spre exemplu, controlului mecanismului de evaluare, poate utiliza o memorie de tip stivă fără a fi necesară comutarea sa în vederea comandării mecanismului de implementare a stivei. Comutarea funcțională neputîndu-se face în timp nul, existența celor două procese dedicate,  $P_0$  și  $P_1$  (pentru controlul stivei), optimizează ansamblul.

Utilizarea structurii duale de procesor prezintă avantaje și în privința minimizării efortului hardware de implementare [6].

În cazul (2) putem spune, pentru început, că paralelismul creează o premisă aducerii la un nivel hardware cît mai coborît a unor funcțiuni de nivel superior. O funcție dată, particulară, implică o structură dedicată. Alocarea unei structuri pentru o funcție se poate face deosebit de eficient într-un sistem paralel de procesare. Paralelismul nu este cu necesitate presupus, dar este de dorit și apariția lui poate fi stimulată și pe această cale.

Această tendință nu este numai generatoare de paralelisme; ea se justifică și prin faptul că realizarea unor funcții la un nivel ridicat creează flexibilitate în implementare dar este foarte inefficientă sub aspectul timpului de execuție. De regulă, implementarea la un nivel superior presupune adecvarea unor mecanisme generale la o problemă particulară. Exemplul cel mai simplu este dat de încercarea de a reduce toate problemele la tratarea prin mecanisme ce presupun numericul ca model esențial. Comparația, spre exemplu, este o funcție logică, dar de cele mai multe ori este realizată ca o secvență de operații aritmetice.

Limbajul LISP cere, în esență, implementarea directă a unor operații foarte simple de natură logică, cum ar fi: comparări, setări de configurații sau de biți independenți, testări simple sau multiple. Realizarea lor pe o mașină universală se dovedește greoaie, inefficientă. O cerință esențială va fi deci aceea de a implementa eficient, direct printr-un hardware adecvat funcții de tipul celor anterior listate.

Această observație impune resursele fizice ale elementelor de procesare.

Nivelurile de implementare în structura DIALISP sînt următoarele: (3) funcții logice în RALU, (4) microprograme, (5) asamblare.

Nivelul (3) este impus de operațiile elementare anterior listate și de circuitele disponibile într-o variantă cît mai compactă. Nivelul (4) este cel ce ar trebui extins cel mai mult. Funcțiile LISP sînt în mare măsură implementate prin microprograme. Setul celor implementate în această manieră este limitat de faptul că memoria de microprograme este o resursă scumpă și dificil de implementat la mari capacități și viteză convenabilă cu circuite ușor disponibile. Această limitare a impus nivelul (5) de implementare.



Din punct de vedere software DIALISP este structurată pe trei niveluri : (6) microprogram, (7) asamblare, (8) LISP.

Luând în considerare poziția mașinii comparativ cu alte mașini LISP similare, memoria de control de dimensiune mică ( $8K \times 72$  biți) a impus structurarea software pe trei nivele.

Experiența cu mașina DIALISP în actuala variantă a arătat că o memorie de control de capacitate cel puțin 16 Kcuvinte ar permite structurarea la două nivele (microprogram și LISP); eventual permițând existența unui nivel intermediar (similar nivelul de asamblare) necesar dezvoltării unui compilator LISP.

Până în momentul de față, dezvoltarea celui de-al doilea nivel (asamblare) s-a făcut în două variante, ceea ce a permis alegerea strategiei optime din punct de vedere al setului de instrucțiuni în asamblare, vis-à-vis de maniera de implementare a evaluării LISP.

Deoarece evaluatorul LISP a fost realizat în limbaj de asamblare, modul de alegere al setului de instrucțiuni este hotărâtor în ceea ce privește performanța (viteza de evaluare).

În prima variantă, pe o structură de stivă ce permitea ambele moduri de evaluare shallow-deep s-a ales o mulțime de instrucțiuni caracterizate prin simetrie și generalitate (referitor la posibilitățile de acoperire a implementării interpretorului la nivelul de asamblare).

Această variantă s-a dovedit deficitară în ceea ce privește viteza în evaluare, dar optimă ca strategie în cazul existenței unei memorii de control de capacitate mică (posibilitatea înglobării la nivel de microprogram a unui set relativ bogat de funcții LISP pure).

A doua variantă a adoptat o soluție deep-binding pentru stivă și un set de instrucțiuni asamblate caracterizat prin asimetrie și orientare spre stiva LISP. S-a ținut seama de frecvența de apariție a anumitor operații, de modul în care funcțiile LISP pure își lasă sau nu, valorile în stivă, precum și de necesitatea existenței unor instrucțiuni care să permită implementarea unor funcții în manieră nerecursivă etc.

### 3. Structura sistemului DIALISP 01

Componentele principale ale mașinii LISP sînt : (9) unitatea RALU de prelucrare a cuvintelor de 32 de biți, (10) automat dual cu stivă pentru controlul întregii structuri (AS), (11) memoriile  $M_0$  și  $M_1$ , organizate pe cuvinte de 32 de biți, (12) interfață paralelă pentru cuplarea pe BUS-ul sistemului DIAGRAM (IP), (13) interfață serială pentru cuplarea pe calea de date cu interfață de disc (IS).

În figura 3 este prezentată schema bloc a sistemului DIALISP 01. Unitatea RALU a fost implementată utilizînd circuite INTEL 3002 cu avantajele, dar mai ales dezavantajele datorate structurii interne restrictive. Alegerea a fost făcută ținînd cont de facilitățile de cuplare oferite de aceste cipuri precum și de existența unor furnizori siguri. Într-o soluție ce se dorea foarte compactă această alegere era de asemenea justificată. Memoriile  $M_0$  și  $M_1$  sînt realizate cu circuite de 64 Kbiți.

Structura principală a AS este prezentată în figura 4. Principiul schemei ca AS este prezentat în [7]. Capacitatea de a controla două procese i-a fost conferită de cele două numărătoare  $N_0$  și  $N_1$  ce adresează MS (memoria de stivă) prin  $MUX_0$  comandat de semnalul  $\Phi$  (figura 2).



Fig. 3. — Structura generală a mașinii DIALISP 01

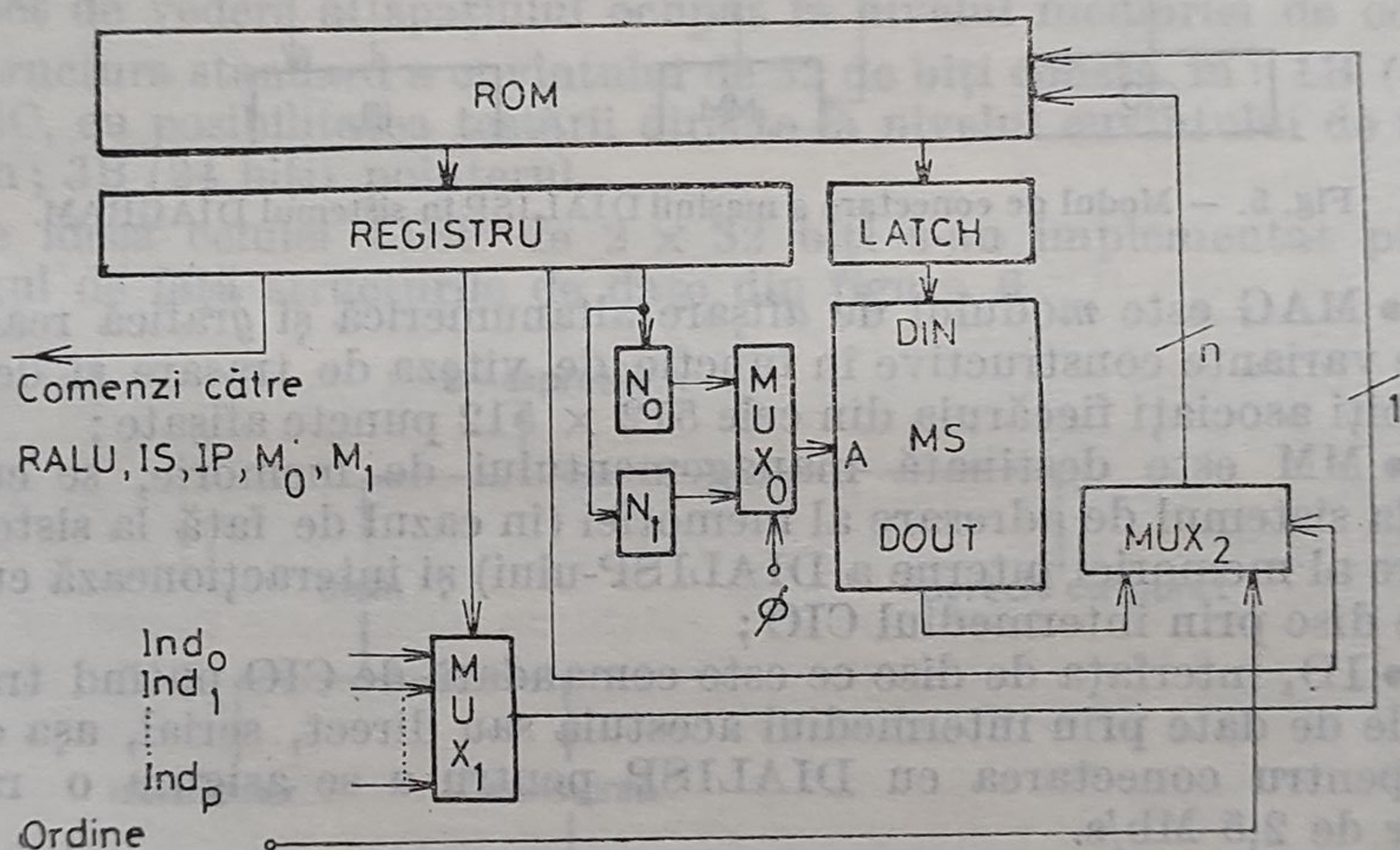
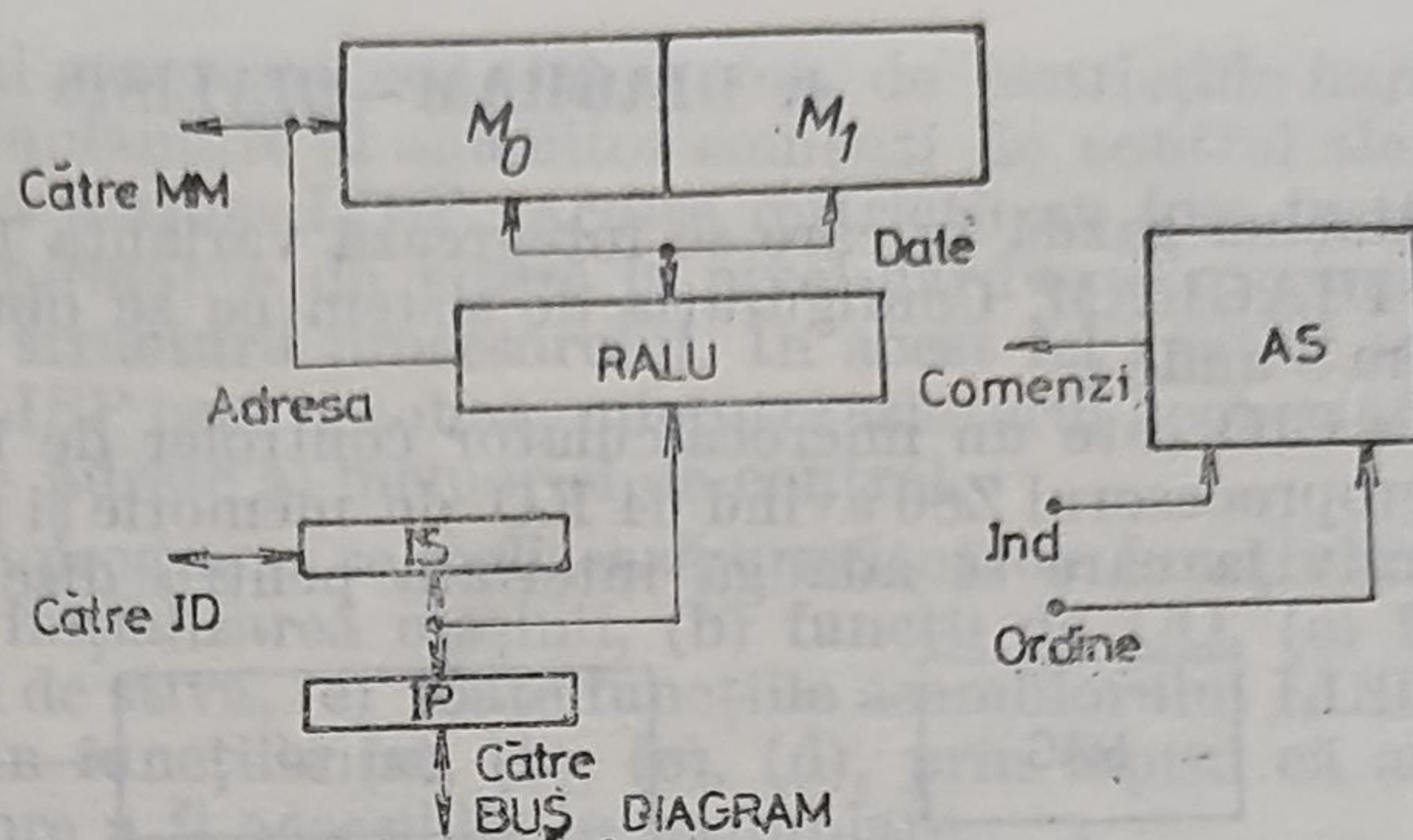


Fig. 4. — Schema automatului de control cu stiva pe buclă.

În această variantă mașina LISP are următorii parametri structurali :  
 memorie internă :  $M_0$  de 128 KW și  $M_1$  de 64 KW (cuvinte de 32 de biți) ;  
 memorie externă :  $4 \times 4$  MO pe disc ; capacitate de adresare :  $1 \div 24$  de biți, deoarece fiecare proces beneficiază de o adresă de 24 de biți ;  
 capacitatea memoriei de microprogram : 8 KW, cu un cuvânt de 72 biți ;  
 stiva asociată memoriei de microprogram : câte 16 niveluri pentru fiecare proces.

Limitările principale ale acestei variante sînt date de structura prea generală a circuitelor 3002 ce configurează unitatea RALU. Adaptarea la structura lor particulară consumă un număr prea mare de cuvinte de microprogram, lungind, în consecință, durata de execuție a funcțiilor. Acest fapt implică și o memorie de microprogram nejustificat de mare. Se ajunge astfel și la limitarea dată de memoria de microprogram.

O structură dedicată în locul unui circuit universal va permite o viteză cel puțin dublă de execuție sub controlul unei memorii de microprogram dimensionată la jumătate sau în condiții în care se realizează prin microprogramare un număr mare de funcțiuni, cu implicații în mărirea vitezei de evaluare.



#### 4. DIAGRAM—DIALISP

Mașina gazdă în care se integrează varianta DIALISP 01 este sistemul DIAGRAM. Configurația de sistem ce se obține este reprezentată în figura 5 unde :

- CIO este un microcalculator controler de intrare-ieșire, realizat cu microprocesorul Z80 avînd 64 KO de memorie și periferie pentru lucrul interactiv la care se adaugă interfața pentru disc flexibil ;

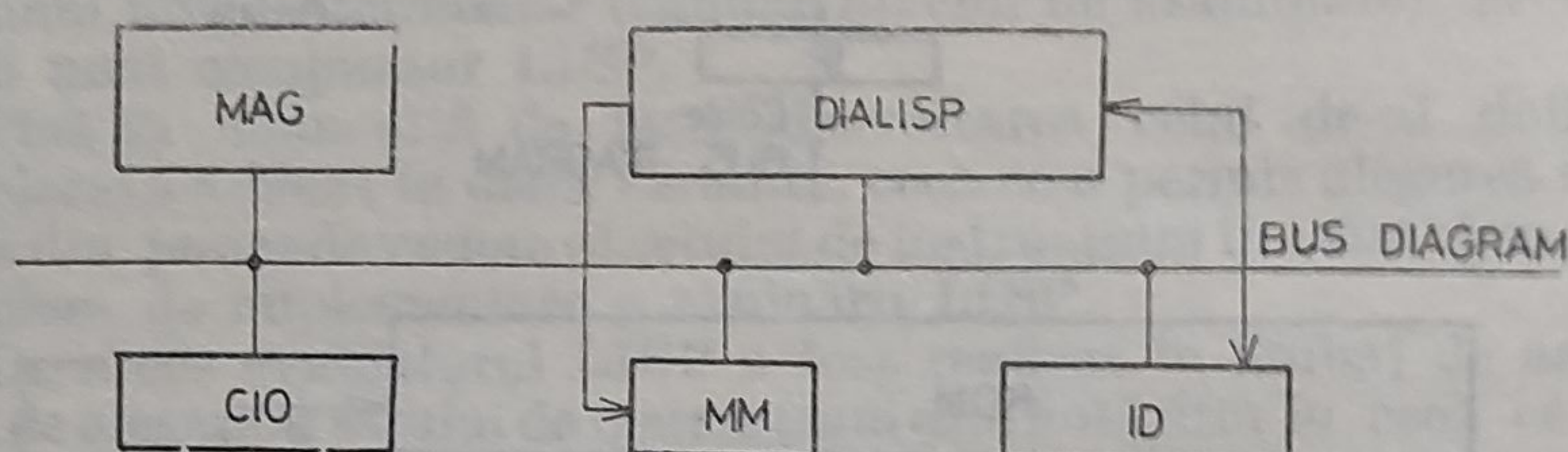


Fig. 5. — Modul de conectare a mașinii DIALISP în sistemul DIAGRAM.

- MAG este modulul de afișare alfanumerică și grafică realizat în diferite variante constructive în funcție de viteza de trasare și de numărul de biți asociați fiecăruia din cele  $512 \times 512$  puncte afișate ;

- MM este destinată managementului de memorie, se cuplează direct la sistemul de adresare al memoriei (în cazul de față la sistemul de adresare al memoriei interne a DIALISP-ului) și interacționează cu interfața de disc prin intermediul CIO ;

- ID, interfața de disc ce este comandată de CIO putînd transfera blocurile de date prin intermediul acestuia sau direct, serial, așa cum s-a optat pentru conectarea cu DIALISP pentru a se asigura o rată de transfer de 2,5 Mb/s.

În această configurație DIALISP 01 este realizat pe două plăci cu circuite integrate.

#### 5. DIALISP 01 — Software

Limbajul LISP pe mașina actuală a fost realizat în două strategii. În prima, mașina LISP este văzută ca o mașină cu regiștri, iar în cealaltă ca o mașină de tip stivă. Cele două variante au ținut cont de posibilitatea existenței a două procese paralele. Astfel spațiul de adrese al celor două memorii interne a fost structurat astfel : memoria  $M_0$  este divizată în două subspații : (a) spațiul codului binar generat de asamblorul LISP, (b) spațiul binar (celule LISP).

Memoria mașinii  $M_1$  este afectată stivei ce implementează evaluarea LISP (control și legarea variabilelor), deci este divizată în două subspații : (a) stiva de control, (b) stiva variabilelor LISP.

În acest fel dualitatea mașinii LISP privește două procese software : cel al funcțiilor LISP pure ce operează pe memoria  $M_0$  și cel al evaluării LISP propriu-zise ce operează pe memoria  $M_1$ . La nivelul de microprogramare s-a folosit ca instrument de dezvoltare un microtranslator realizat inițial în DM-LISP pe CORAL, I 100, apoi reseris pe procesorul CIO (Z 80) al sistemului DIAGRAM 2030.



Microtranslatorul generează cod ținând cont de restricțiile hardware relative la modul de implantare al anumitor comenzi de control ale automatului ce gestionează mașina LISP. Aceste restricții au fost justificate de obținerea unor performanțe de viteză la nivel hardware precum și de spațiul limitat alocat structurii procesorului. În acest fel spațiul stărilor automatului mașinii LISP este neconex, microtranslatorul acoperind însă, în final, tot spațiul de adrese al memoriei de control.

La nivel de microprograme se realizează următoarele funcții-mașină : (a) funcții relativ la inițializarea mașinii, (b) funcții de I/O, (c) funcții LISP pure, (d) funcții de stivă, (e) toate funcțiile asamblorului LISP care înglobează o parte din funcțiile (a), (b), (c), (d), prin faptul că au fost scoase la suprafață spre a fi accesibile în asamblare.

Ca o remarcă funcțiile de I/O s-au dovedit a fi foarte costisitoare din punct de vedere al spațiului ocupat la nivelul memoriei de control.

Structura standard a cuvîntului de 32 de biți constă în : 1B (8 biți) tip + GC, cu posibilitatea testării directe la nivelul cuvîntului de microprogram ; 3B (24 biți) pointerul.

Pe ideea celei LISP de  $2 \times 32$  biți s-au implementat pînă în momentul de față structurile de date din figura 6.

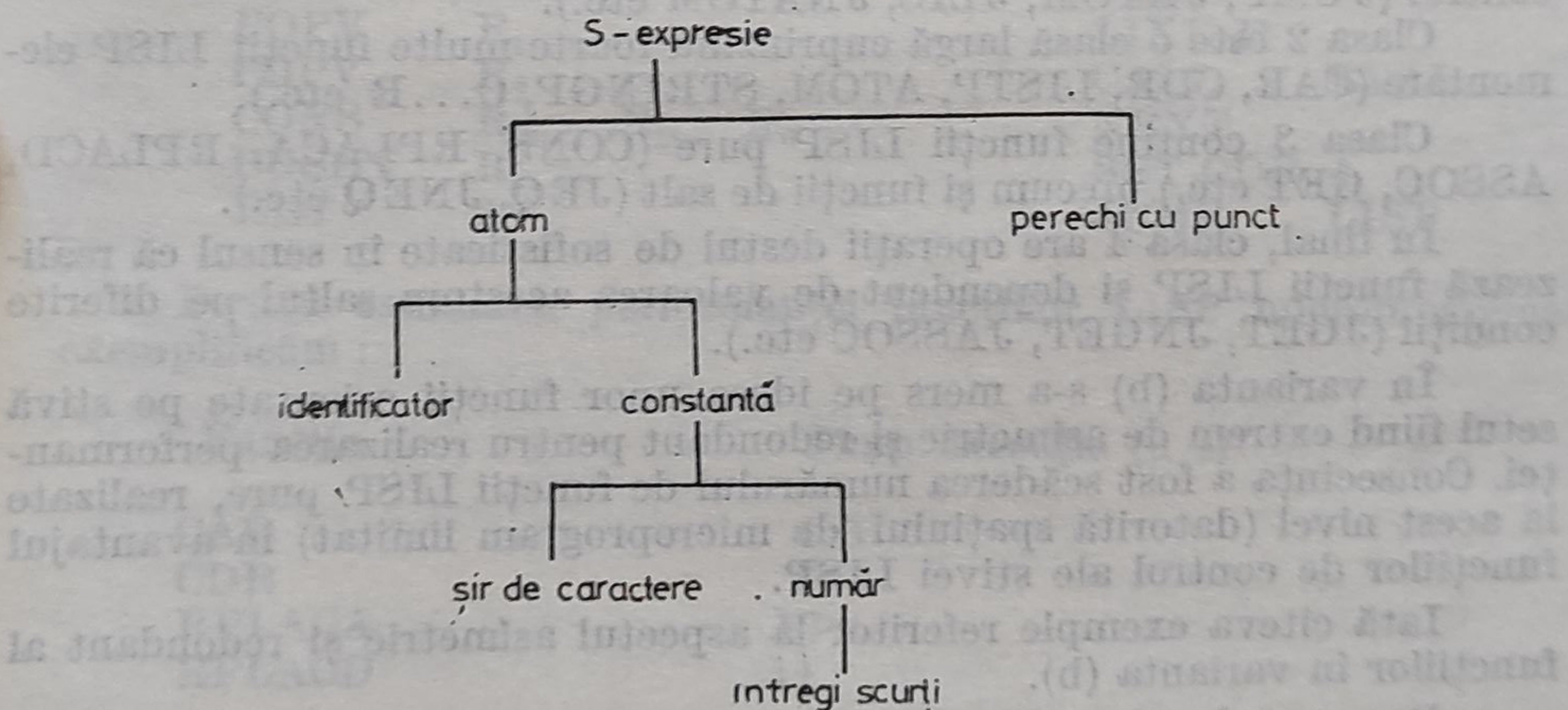


Fig. 6. — Structuri de date.

Trebuie remarcat ca un lucru deosebit de important modul în care este folosită dualitatea mașinii la nivelul microprogramelor. Pentru că cele 2 procese — cel al funcțiilor LISP pure și cel din stivă — nu sînt perfect simetrice, s-a oferit posibilitatea hardware de a se elimina timpul mort care s-ar pierde în situația în care unul din procese devine dominant (un astfel de exemplu este căutarea valorii unei variabile pe stivă, moment, în care procesul asociat stivei LISP ( $P_1$ ) pune în așteptare procesul  $P_0$  care așteaptă valoarea). Acest lucru este rezolvat prin furtul ciclului procesului  $P_0$  pe care  $P_1$  îl folosește, scurtînd astfel durata căutării.

Situația este rezolvată similar în cazul în care procesul  $P_0$  devine dominant.

Nivelul de asamblare este cel care permite implementarea evaluato-  
rului și a funcțiilor LISP sistem. S-a construit în acest sens un asamblor



într-o singură trecere, realizat în prima variantă în DM-LISP, apoi în varianta finală pe procesorul CIO (Z 80) al sistemului DIAGRAM 2030.

În momentul de față se lucrează la un asamblor în două treceri care să rezolve problema generării codului pe un spațiu oricât de mare.

Modul în care s-au ales instrucțiunile la acest nivel a fost hotărât în ceea ce privește performanța mașinii.

Codul generat este încărcat în memoria  $M_0$  și la inițializarea mașinii controlul este transferat buclei READ-EVAL-PRINT.

Instrucțiunile în asamblare au fost concepute ca două seturi distincte relativ la cele două moduri de realizare ale mecanismului de evaluare: (a) mașina cu regiștri, (b) mașina de tip stivă.

Porțiunea de cod de operație (op-code) este de 8 biți în ambele variante. Operanzii pot fi adrese, regiștri (în număr de 5  $R_0-R_4$ ) sau constante (NIL, QUOTE, LAMBDA etc.).

În varianta (a) instrucțiunile sînt împărțite în 5 clase funcție de numărul de operanzi pe care îl au (clasele sînt numerotate de la 0 la 4).

Clasa 0 conține instrucțiuni fără operanzi, ca de exemplu TERPRI, RATOM etc.

Clasa 1 conține funcții LISP pure (PATOM, PATOMC etc.) precum și funcții de stivă (CALL, RET, PUSHN etc.), de asemenea, funcții de control (JUMP, JATOM, JNIS, JNATOM etc.).

Clasa 2 este o clasă largă cuprinzînd foarte multe funcții LISP elementare (CAR, CDR, LISTP, ATOM, STRINGP, C...R etc.).

Clasa 3 conține funcții LISP pure (CONS, RPLACA, RPLACD, ASSOC, GET etc.) precum și funcții de salt (JEQ, JNEQ etc.).

În final, clasa 4 are operații destul de sofisticate în sensul că realizează funcții LISP și dependent de valoarea acestora saltul pe diferite condiții (JGET, JNGET, JASSOC etc.).

În varianta (b) s-a mers pe ideea unor funcții orientate pe stivă setul fiind extrem de asimetric și redondant pentru realizarea performanței. Consecința a fost scăderea numărului de funcții LISP pure, realizate la acest nivel (datorită spațiului de microprogram limitat) în avantajul funcțiilor de control ale stivei LISP.

Iată cîteva exemple referitor la aspectul asimetric și redondant al funcțiilor în varianta (b).

De exemplu, funcțiile de salt pe tipurile de date se realizează în două moduri:

Mnemonică-test deplasament, adresă-salt

sau

Mnemonică-test, adresă-salt,

în prima situație se testează conținutul slot-ului pe stiva LISP (dat de adîncimea în stivă prin deplasament) efectuîndu-se saltul pe condiția testată, în a doua situație testul se face pe unul din regiștrii interni ai RALU (folosit pentru comunicare între cele două procese  $P_0$  și  $P_1$ ) care conține în general valori întoarse de funcțiile LISP (procesul  $P_0$ ). Din acest exemplu se observă că funcțiile LISP pure depun valorile într-un registru de comunicare, dar mașina fiind orientată spre stivă aceste valori sînt depuse corespunzător pe ea. Acest lucru nu se întîmplă însă decît atunci cînd este necesar, unele funcții — cum este CONS — aruncînd valoarea în stivă mereu, altele nefiind necesară o trimitere explicită către slot-ul din stivă.



În final enumerăm câteva funcții asamblate în varianta (b) :

- teste pe tipuri JATOM, JLIS, JNSTR, JREC etc.;
- funcții LISP : C...R, CONS, NCONS, RPLACA, RPLACD etc.;
- funcții pe stivă : CALL, RET, LINK-D, SAVE-ED, SET-CDR, SET-CAR ALLOC, ALLOCN etc.

Vom da două exemple pentru a arăta modul în care se scrie aceeași funcție în asamblare, conform variantelor (a) și (b). Funcția este LIST și construiește o listă evaluându-și argumentele unul câte unul :

#### VARIANTA (a) (recursiv)

LIST	JT	R <sub>1</sub> ,	LST <sub>1</sub>
	PUSHV	R <sub>1</sub>	
	RET	2	
LST <sub>1</sub>	PUSHN	2	
	CAR	R <sub>1</sub> , R <sub>1</sub>	
	CALL	EVAL	
	PUSHN	2	
	CDR	R <sub>1</sub> , R <sub>1</sub>	
	CALL	LIST	
	POPV	R <sub>1</sub>	
	POPV	R <sub>2</sub>	
	CONS	R <sub>2</sub> , R <sub>1</sub> , R <sub>0</sub>	
	PUSHV	R <sub>0</sub>	
	RET	2	

#### VARIANTA (b) (nerecursiv)

LIST	JNEQ	4, NIL, LST <sub>1</sub>
	LINK-D	
	NCONS	
	REST-E	
	RET	
	SAVE-ED	
	ALLOCN	1, EVAL
	CAR	4
	SEND	
	LINK-D	
	CALL	EVAL
	NEXT	
	SET-CDR	4
	JUMP	LIST

Pentru estimarea performanței funcțiilor LISP microprogramate exemplificăm :

Funcție	Timp execuție (μ s)
CAR	4.5
CDR	5
RPLACA	12
RPLACD	14

## 6. Structura sistemului DIALISP 02

Cea de a doua variantă a sistemului DIALISP păstrează, în mare, configurația primeia, înlăturând limitările impuse sub aspect cantitativ mașinii DIALISP 01.

Astfel, nu se mai optează pentru un RALU realizat cu circuite bit-slice, cu consecința evidentă a conceperii unei structuri ce utilizează mai multe CI dar configurează o structură mult mai eficientă. De asemenea se va atașa sistemului o capacitate de memorie locală de 1 MW (cu un cuvânt de 32 de biți), oferind o memorie externă pe disc mai mare de 128 Mocteti.

O altă schimbare, aparent cantitativă numai, este extinderea stivei asociate automatului de control AS (vezi figura 4). În cazul în care numărul de nivele ar depăși  $0,5 \div 10^6$ , atingând maximum  $2 \div 10^6$ , se poate pune problema microprogramării funcției EVAL obținându-se o



mașină deosebit de performantă, la limita permisă de timpul de acces la memoria sistemului.

Dacă la acestea se adaugă mărirea capacității memoriei de microprogram ( $16 \div 64$  KW) ale cărei instrucțiuni comandă în secvențe mult mai eficiente resursele sistemului, se creează condițiile *dispariției nivelului funcțional al asamblării*, cu consecințele evidente asupra vitezei de lucru a sistemului.

La toate acestea se va adăuga faptul că procesul  $P_1$ , cel ce gestionează stiva de argumente, va beneficia de accesul la primele  $250 \div 1\,024$  locații din stivă dintr-o memorie TTL rapidă. Nu se poate utiliza o memorie rapidă similară și pentru procesul  $P_0$  deoarece înlănțuirea în  $K_0$  este oarecare, pe traiectorii imprevizibile.

Se prevede creșterea vitezei de execuție a programelor scrise în LISP de  $3 \div 6$  ori în cazul reconfigurărilor propuse. Menționăm însă că pentru DIALISP 02 vor fi necesare  $4 \div 6$  astfel de plachete de circuit imprimat în funcție de performanțele dorite. În varianta maximă, trebuie menționate cele peste 1 000 de memorii de 64 Kbiți necesare, ce vor ocupa 4 din cele 6 plachete cu CI. O plachetă va fi destinată automatului de control (AS), iar ultima unei unități LUR (Logic Unit and Registers) ce va fi implementată cu circuite standard MSI și LSI.

Înlocuirea RALU, realizată cu circuite universale, cu LUR configurat cu circuite standard, permite adecvarea unității de prelucrare la funcțiile limitate, dar precis orientate presupuse de implementarea limbajului LISP. LUR este format dintr-o unitate pe 32 de biți, una de 24, una pe octet și una pe bit, permițând realizarea rapidă a testărilor pe cuvânt, adresă sau bit, a resetărilor pe aceleași configurații precum și a operațiilor pe șiruri de caractere.

Limitarea esențială ce mai apare în cazul variantei DIALISP 02 este cea dată de capacitatea de memorie de microprograme care nu poate fi extinsă la dimensiuni prea mari din cauza consumului mare presupus de memoriile rapide.

Consecințele adaptării variantei DIALISP 02 din punct de vedere software ar fi mărirea vitezei de execuție.

Evaluatorul implementat în întregime la nivel microprogramat ar presupune mărirea considerabilă a vitezei precum și dispariția, la limită, a codului binar generat de asamblorul LISP.

Asamblorul LISP (regândit în această variantă) se justifică în cazul realizării unui compilator.

Pe de altă parte, posibilitatea abordării la nivel de microprogram ar presupune un mecanism de garbage-collection foarte rapid.

În sfârșit, memoria internă de 1 M cuvânt ar oferi posibilitatea unei bune performanțe pentru programele aplicative la care s-ar diminua pierderea în timp datorată acceselor la disc, mult mai frecvente în situația unei memorii interne mici.

## 7. Varianta DIALISP 11

Am văzut că limitarea esențială a sistemului DIALISP 02 este impusă tehnologie memoriei de microprogram. Parametrul care impune indirect limitarea este timpul de acces ce se dorește foarte mic, de ordinul  $30 \div 100$  ns. Pentru a înlătura această restricție, unitățile co-



mandate de AS ar trebui să efectueze o microinstrucțiune într-un interval de timp mai mare, fără ca acest fapt să afecteze viteza sistemului. Afirmatia anterioară, aparent paradoxală, își poate găsi un corespondent într-o mașină în care, spre exemplu, fiecare acțiune a LUR presupune un acces la memoria  $M_0$  sau  $M_1$ , sau în cazul în care la nivelul LUR se poate defini o secvențare locală de  $2 \div 4$  tacturi pentru efectuarea oricărei microinstrucțiuni.

Aceste restricții pot fi relativ ușor satisfăcute ținând cont de natura limbajului LISP ce nu presupune prelucrări complexe asupra operanzilor, ci manipularea unui număr mare de date stocate în memorie. De asemenea, această opțiune pentru memoria de microprogram presupune realizarea unei unități LUR mai complexe cu un eventual control local minimal. Vom complica însă o resursă ce nu a impus restricții, în favoarea simplificării unei resurse deficitare.

Aceste opțiuni presupun un oarecare risc prin particularizările impuse. Minimizarea acestui risc se poate face însă ca urmare a unei experiențe în implementarea hardware a limbajului LISP. Din acest motiv considerăm o astfel de variantă ca ulterioară celor deja prezentate.

Schema de principiu a unei astfel de mașini ar putea fi cca reprezentată în figura 7, unde:

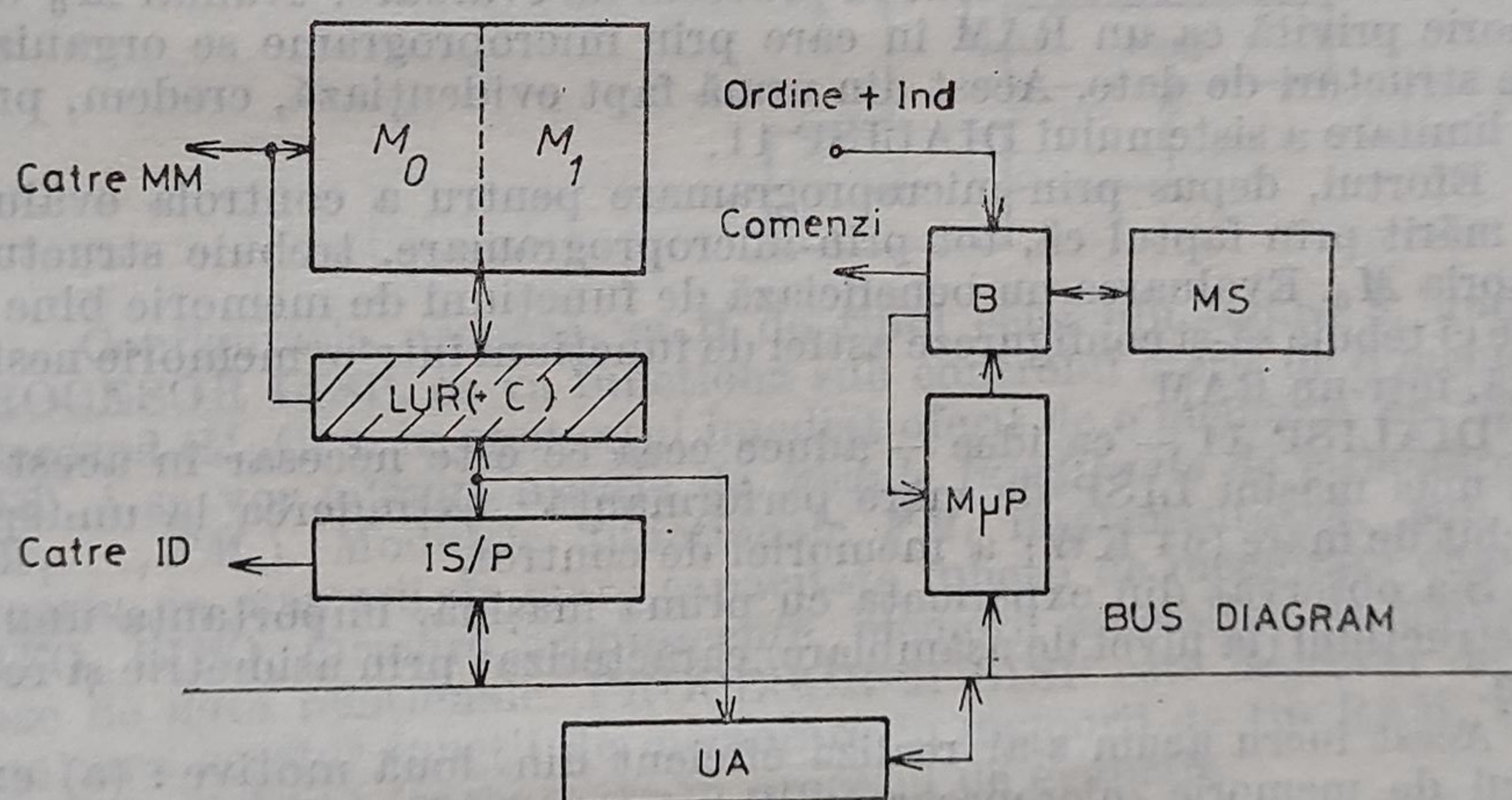


Fig. 7. — Structura sistemului DIALISP 11.

—  $M_0$  și  $M_1$  sînt memoriile asociate celor două procese  $P_0$  și  $P_1$ ; într-una se dezvoltă stiva de argumente ( $M_1$ ) iar în cealaltă se organizează restul structurilor de date;

— IS/P este o interfață serială și paralelă pentru cuplarea lentă la READ și PRINT (pe interfața paralelă) sau rapidă, pentru comunicarea cu discul pe cea serială;

—  $M_{\mu P}$  este memoria de microprogram de 64 KW realizată cu circuite MOS cu ciclul de  $250 \div 500$  ns;

— MS este o memorie de tip stivă pentru salvarea adreselor de retur în EVAL;

— B cuprinde circuitele ce asigură închiderea buclei automatului cu stivă (AS) format de  $M_{\mu P}$ , MS și B;

— LUR(+C) reprezintă componenta ce dă specificitate acestei configurații, fiind formată din LUR și eventual un modest control local rapid;



— UA este o unitate aritmetică destinată să preia toate operațiile cu caracter numeric ce ar apare în evaluare; reprezintă un modul universal și independent ce poate fi implantat opțional.

Memoria de microprogram poate fi încărcată de pe BUS-ul sistemului DIAGRAM, asamblarea microinstrucțiunilor fiind realizată de CIO (fig. 5).

Această variantă nu va fi o mașină ce va avea o viteză hardware mai mare decât variantele anterioare, dar viteza de execuție va putea crește notabil datorită posibilității de a înlătura definitiv nivelul de asamblare. De asemenea, implementarea fizică va fi mai comodă datorită faptului că  $M_{\mu P}$  va înlocui o structură mare consumatoare de putere cu una având un consum nesemnificativ în ansamblu.

Remarcăm faptul că DIALISP 11 se prezintă ca o *rețea de memorii* interconectate printr-un *hardware minimal* în contextul celor aproximativ 1000 cipuri de memorie utilizate. Dintre acestea  $M_0$  și  $M_1$  se pot extinde pe disc iar MS și  $M_{\mu P}$  sînt limitate la structura implantată local. Se remarcă și o parțială specializare funcțională a acestor memorii. Astfel:  $M_1$  este destinată stivei de argumente organizată sub controlul AS prin intermediul LUR (+C); MS este o stivă pentru adrese de retur, limitată și controlată printr-un mecanism dedicat;  $M_{\mu P}$  simulează un ROM, conținutul ei neputînd fi modificat în procesul de evaluare. Numai  $M_0$  este o memorie privită ca un RAM în care prin microprograme se organizează unele structuri de date. Acest din urmă fapt evidențiază, credem, principala limitare a sistemului DIALISP 11.

Efortul, depus prin microprogramare pentru a controla evaluarea este mărit prin faptul că, tot prin microprogramare, trebuie structurată memoria  $M_0$ . Evaluarea nu beneficiază de funcțiuni de memorie bine precizate ci trebuie să-și configureze astfel de funcțiuni într-o memorie nestructurată, într-un RAM.

DIALISP 11 — ca idee — aduce ceea ce este necesar în acest moment unei mașini LISP de mare performanță: extinderea la un spațiu deosebit de mare (64 KW) a memoriei de control.

S-a observat din experiența cu prima mașină, importanța unui set de instrucțiuni (la nivel de asamblare) caracterizat prin asimetrie și redundanță.

Acest lucru acum s-ar realiza eficient din două motive: (a) crește spațiul de memorie microprogram, (b) funcțiile la nivelul asamblării coboară la limită la nivelul microprogram.

Mașinile LISP realizate în S.U.A. au avut în faza inițială memorii de control de 16 KW ca să ajungă la capacități uriașe de 64 KW (ROM/RAM).

Mărirea la aceste dimensiuni este condiționată tehnologic dar experiența arată că acesta este modul în care evoluția performanței poate avea loc în această configurație.

## 8. DIALISP 21

Una din caracteristicile fundamentale ale mașinilor digitale clasice, gîndite ca fiind structurate printr-o zonă de procesare și una de memorare, este aceea că *memoria este structurată prin intermediul unui mecanism rulat tot de procesor*. Efortul de procesare este deci mărit prin faptul că memoria atașată procesorului este *nul structurată*.



Soluțiile anterior prezentate pentru sistemul DIALISP nu reușesc să îndepărteze cu totul această deficiență primordială. A o înlătura, pe lângă avantajele ce se întrevăd, reprezintă și un risc ce nu poate fi asumat decât după o experiență suficientă, sau în condițiile în care se poate asigura o simulare avansată a structurii propuse. Deoarece nu întrevădem posibilitatea unor simulări edificatoare pe mașini universale, o astfel de abordare va urma, chiar dacă nu strict, experiențelor preliminare făcute cu DIALISP 01, 02 și 11.

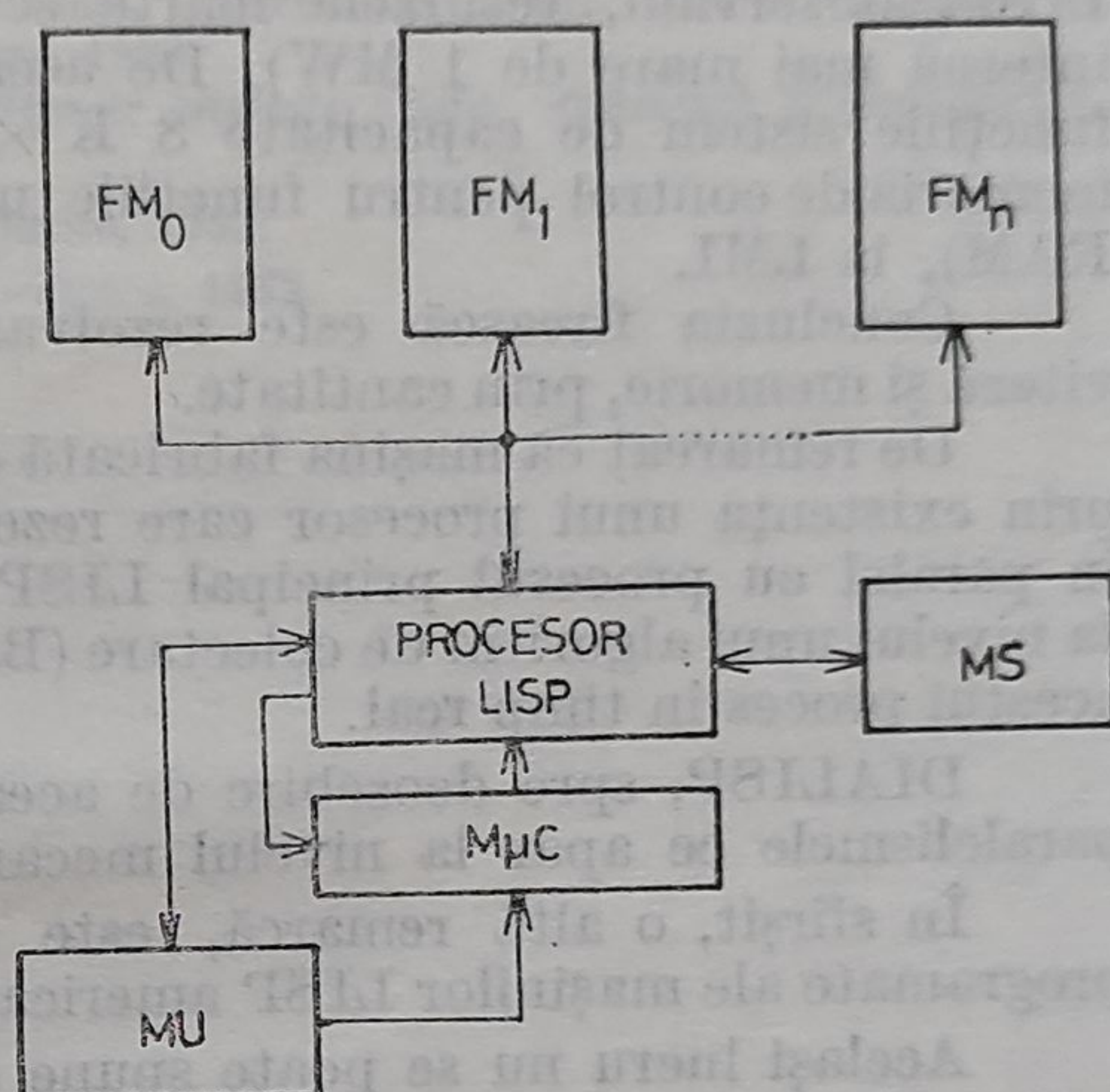


Fig. 8. — Implementarea unei mașini LISP utilizând memorii funcționale (DIALISP 21).

Configurația propusă va fi de tipul celei din figura 8, unde unui PROCESOR LISP, ce va funcționa sub controlul oferit de o memorie de microcod ( $M_{\mu}C$ ) și în contextul imediat oferit de o memorie de tip stivă, ( $MS$ ), i se vor adăuga o serie de *module funcționale de memorare* ( $FM_0$ ,  $FM_1, \dots, FM_n$ ). Modulele funcționale  $FM_i$  implementează, sub control propriu, în memorii de mare capacitate funcții de memorare cum ar fi: FIFO, LIFO, memorii arborescente, memorii asociative, la limită chiar baze de date relaționale. PROCESOR-ul LISP este degrevat de implementarea acestor funcții de memorare în memorii de tip RAM, obținându-se o eficiență foarte mare în procesul de evaluare.

Dacă ansamblul PROCESOR LISP + MS +  $M_{\mu}C$  poate constitui o configurație compactă conectabilă comod la o mașină universală ( $MU$ ) modulele  $FM_0, FM_1, \dots, FM_n$  pot constitui fiecare mașini mai mult sau mai puțin complexe avînd și altă destinație decât integrarea lor într-o mașină LISP. Într-o astfel de structură nu mai putem spune că avem o unitate de procesare și una de memorare, deoarece funcția de procesare se extinde și în modulele de memorie iar memoria se fragmentează într-o rețea în care rolul fiecărei secțiuni depinde de o structură hardware de procesare dedicată local.

## 9. Alte mașini LISP

Preocupările în sensul proiectării mașinilor LISP s-au născut în S.U.A. începînd cu anul 1973.

Lăsînd deoparte implementările single-chip care privesc alte clase de procesare LISP (realizarea lor fiind strîns legată de tehnologia folosită)



putem face o comparație în special la nivel de concepție între DIALISP și alte mașini din clase apropiate.

Astfel, cele două mașini LISP disponibile comercial în S.U.A. la începutul anilor '80 sînt fabricate de LMI (LISP Machine Incorporated) și Symbolics Inc.

Cele două mașini sînt derivate din mașina CADR proiectată la MIT.

Problemele importante fiind cele ce privesc proiectarea procesorului LISP, observăm, resursele foarte scumpe puse la dispoziție (memorie internă mai mare de 1 MW). De asemenea, memorii de control pentru funcțiile sistem de capacitate  $8\text{ K} \times 112$  biți (ROM), la Symbolics și memoria de control pentru funcțiile utilizator microcompilate ( $64\text{ KW/RAM}$ ), la LMI.

Concluzia firească este rezolvarea problemelor LISP-ului, adică viteză și memorie, prin cantitate.

De remarcat că mașina fabricată de LMI dispune de procesare duală prin existența unui procesor care rezolvă colectarea spațiului disponibil în paralel cu procesul principal LISP. Paralelismul este deci exploatat la nivelul unui algoritm de colectare (Baker 1977) care permite efectuarea acestui proces în timp real.

DIALISP, spre deosebire de această abordare, caută să surprindă paralelismele ce apar la nivelul mecanismelor proprii limbajului LISP.

În sfîrșit, o altă remarcă, este arhitectura procesoarelor micro-programate ale mașinilor LISP americane, care este una clasică.

Același lucru nu se poate spune despre DIALISP la care anumite opțiuni hardware la nivelul controlului realizat printr-un automat cu stivă, îl deosebește ca idee de celelalte două mașini.

În sfîrșit, existența și în DIALISP a unor posibilități hash-hardware ar îmbunătăți considerabil performanța anumitor aplicații. S-ar rezolva problema căutărilor costisitoare, problemă rezolvată în proiectarea mașinii LISP japoneze FLATS [5].

Concluzia finală a acestei sumare treceri în revistă este aceea a sensului în care trebuie să evolueze procesoarele LISP și anume al apropierii lor de limbaj (de mecanismele acestuia), această apropiere fiind necesar a fi privită ca una funcțională dar nu rigidă ci beneficiind de o anumită mobilitate.



## BIBLIOGRAFIE

1. John BACKUS, *Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs*, Communications of the ACM, **21**, 8, August (1978).
2. John BACKUS, *Function-level Computing*, IEEE Spectrum, August 1982.
3. Allen BAWDEN ș.a., *Lisp Machine Progress Report*, 444 Memo, Artificial Intelligence Laboratory, MIT, 1977.
4. L. Peter DEUTSCH, *Experience with a Microprogrammed Interlisp System*, Xerox Palo Alto Research Center, 1979.
5. E. GOTO ș.a.; *Design of a Lisp Machine-Flats*, Proceedings of the ACM Symposium on Lisp and Functional Programming, 1982.
6. M. Gheorghe ȘTEFAN ș.a., *Circuite integrate digitale*, Edit. didactică și pedagogică, București, 1983.
8. \* \* \* *Overview of LMI Lisp Machine Software*, 1982.
9. \* \* \* *Overview of LMI Lisp Machine Hardware*, 1982.



## POSTFAȚĂ

Tehnica electronică de calcul s-a dezvoltat în numai patru decenii cu o rapiditate care o face de neegalat printre alte ramuri industriale intens tehnologice. Patru generații de calculatoare electronice s-au succedat într-un ritm alert, determinat atât de progresele din domeniul componentelor electronice, al microelectronicii cum se spune în ultimul timp, cât și de progresele înregistrate în domeniul arhitecturii și structurii calculatoarelor electronice și a sistemelor de programe sau software-ului.

Se poate afirma că impactul utilizării calculatoarelor electronice a fost chiar mai mare decât acela produs de conceperea și fabricarea lor industrială. A devenit evident încă din anii '50 că odată cu calculatoarele electronice a început o nouă eră industrială. Comparînd-o cu mașinismul secolelor XVIII și XIX, teoreticienii dezvoltării industriale și economice vorbesc aproape unanim despre o nouă revoluție industrială, care reconfigurează structurile industriale prin introducerea generalizată a elementelor miniaturizate microelectronice și a sistemelor de programe de calculator asociate. Informația numerizată sau digitalizată, în opoziție cu cea avînd purtător analogic, revoluționează modul clasic de colectare, înmagazinare, procesare și distribuire, devenind mai simplu de prelucrat în această nouă formă, cu ajutorul unor calculatoare devenite generație după generații mai potente, mai ieftine, mai ușor de utilizat.

De regulă, cele patru generații de calculatoare au fost definite după apariția calculatoarelor sau cel mult în cazul celei de a patra generații simultan cu apariția unor modele de calculatoare care au reclamat facilități ale respectivei generații.

Fiecare generație a fost definită atât prin progresele înregistrate de circuitele electronice de bază, cu care au fost concepute calculatoarele respectivelor generații, adică tuburile electronice, semiconductoarele discrete, circuitele integrate și circuitele integrate pe scară largă (LSI), cât și prin progresele arhitecturale și de programare care au însoțit evoluția de la structurile simple programate în limbaj mașină ale primei generații la limbajele evolute ale generației a doua și sistemele de operare continue cu elementele de interactivitate, rețele și grafică ale generațiilor a treia și a patra.

Devenită cheia de boltă a noii revoluții industriale, tehnica electronică de calcul a intrat în centrul preocupărilor nu numai a specialiștilor în conceperea, fabricarea și utilizarea calculatoarelor, ci și a unui cerc extrem de larg de oameni interesați în uriașul impact asupra societății, a acestui nou domeniu. Electronizarea, avînd ca principalii suportați tehnica de calcul și microelectronica, a demonstrat rapid că este o soluție exigentă pentru marile probleme ce confruntă orice economie: ridicarea productivității muncii, reducerea consumurilor de materii prime și materiale, energie și combustibil.



Tehnica electronică de calcul a devenit o preocupare pentru majoritatea țărilor lumii care au adoptat programe de dezvoltare corespunzătoare. Pe de altă parte, pe piața mondială capitalistă s-a accentuat concurența între giganții domeniului, în principal din S.U.A. și Japonia, care au influențat în mod indiscutabil scindarea calculatoarelor în patru mari clase: calculatoare mari (numite în literatura de specialitate și main-frames), minicalculatoare, microcalculatoare și mai nou și calculatoarele individuale sau personale.

În acest context a apărut inițiativa japoneză de a defini și eșalona în timp realizarea celei mai noi generații de calculatoare, cea de a cincea.

Propunerea, după inițiatorii ei, urmărește să conducă la o nouă bază teoretică și o nouă tehnologie care să satisfacă necesitățile anilor '90. Pentru prima dată, prin urmare, nu mai asistăm pasivi la dezvoltarea în continuare a domeniului calculatoarelor, ci se încearcă definirea unor obiective ambițioase cu 10—15 ani înainte și se propune o organizare internă în Japonia și o cooperare internațională pentru realizarea integrată a acestor obiective. Programul japonez, anunțat și adoptat în 1981, prevede trei etape:

Etapa inițială 1982—1984 în care să se dezvolte elementele teoretice și tehnologice de bază.

Etapa intermediară 1985—1988, în care să se dezvolte subsisteme, și în fine

Etapa finală, 1989—1991, în care să se dezvolte sistemul complet.

Propunerea, însoțită de sprijinul guvernului japonez, s-a concretizat prin înființarea unui institut de specialitate (Institute for New Generation Computer Technology) și organizarea mai multor conferințe internaționale în care s-au dezbătut principalele direcții de dezvoltare, presupus a fi caracteristice celei de a cincea generații. Ultimele conferințe internaționale au avut loc în 1984 și au fost consacrate și evaluării progreselor înregistrate în etapa inițială.

Inițiativa japoneză a fost primită în mod foarte variat. De la apreciere totală și adoptarea în mai multe țări de măsuri urgente de recuperare a ceea ce se constata a fi avansul japonez, pînă la considerarea, de către unele cercuri de specialitate din S.U.A., ca o simplă ofensivă de marketing, destinată să impună încă și mai mult calculatoarele japoneze pe o piață cu o concurență acerbă. Dat fiind interesul privind impactul în societate al calculatoarelor, dezbaterile a ajuns rapid și în afara cercurilor propriu-zise de specialiști, stimulată și de o carte de succes (Feigenbaum și McCorduck nr. 42/1984). Eliberîndu-ne de caracterul emoțional al dezbaterii, deoarece temă și înainte de anunțarea proiectului japonez, și la fel de puțin important pentru etapa actuală este care țară are poziția de lider, trebuie să constatăm că lansarea proiectului celei de a cincea generații are meritul incontestabil de a fi pus în dezbaterile unor cercuri largi de specialiști și nu numai de specialiști stricți ai domeniului, direcțiile de dezvoltare viitoare a calculatoarelor electronice și etapizare a realizării acestora. Și în țara noastră aceste direcții au reprezentat subiectul unei ample dezbateri organizate în primăvara anului 1984 de Academia R.S. România cu sprijinul specialiștilor din industrie și al Institutului Central pentru Informatică. Preocupări similare se regăsesc și în alte țări socialiste.



De fapt, care sînt principalele idei ale proiectelor de calculatoare din generația a cincea este greu de rezumat într-un spațiu restrîns și ne vom limita la unele aspecte de bază.

Calculatoarele generației a cincea se propun să fie dezvoltate pentru a prelucra informația sub formă de cunoștințe științifice-tehnice sau de altă natură. Pînă în prezent, elementul major al proiectării unui nou calculator era costul circuitelor sau al hardware-ului. Calculatoarele din primele patru generații se bazează în principal pe prelucrarea informației numerice pe *calcule* deci, de unde vine și numele de calculator, pe conceptul de prelucrare secvențială a unor date și instrucțiuni.

O analiză critică a situației actuale ne conduce la concluzia că progresele circuitelor integrate pe scară foarte largă (VLS) au redus costurile circuitelor și se pot implementa eficient toate funcțiunile necesare într-un calculator. De asemenea, constatăm că ne lipsesc funcțiuni de bază pentru prelucrări nenumerate ale textelor, graficelor și vorbirii, că de fapt funcțiunile de bază ale inteligenței artificiale nu sînt în prezent implementate hardware. Una din limitările majore ale calculatoarelor actuale este costul software, care ocupă un procent din ce în ce mai mare din costul sistemului și crează dificultăți în lărgirea sferei aplicațiilor.

Generația a cincea va trebui, în consecință, să răspundă acestor limitări și să soluționeze eficient problemele comunicației om-calculator, a introducerii în această comunicație a inteligenței artificiale.

Proiectul japonez își propune realizarea unui sistem cu patru categorii principale de funcții :

a) Funcțiile de inferență logică și de rezolvare de probleme. Implementarea unor astfel de funcții prin mecanisme de arhitectură distribuită cu prelucrare paralelă va conduce spre sisteme expert cu raționamente deductive și inductive, cu decizii bazate pe date incomplete și alte raționamente logice, caracterizînd pînă în prezent numai sisteme complexe unicat de inteligență artificială.

b) Funcțiile de acumulare și prelucrare de cunoștințe. Sistemele expert necesită acumularea și prelucrarea informației generalizate, sub formă de cunoștințe științifice. Se prevede, în primul rînd, realizarea unor noi arhitecturi de memorii și structuri software corespunzătoare.

c) Funcții de interfață inteligentă. Pentru îmbunătățirea calitativă a interfeței om-calculator, pentru o comunicație flexibilă și „prietenoasă” cu utilizatorul se prevede dezvoltarea de echipamente și programe care să permită reprezentarea și transferul informației grafice, sub formă de imagini, sau sonoră.

d) Funcții de programare inteligentă. Actualul efort de programare ce însoțește realizarea oricărui nou tip de calculator și în continuare utilizarea lui, va fi redus la generația a cincea prin facilități de generare automată a programelor, în primul rînd prin realizarea de sisteme de programare-modulare, automatizarea validării programelor și noi limbaje de programare care să faciliteze automatizarea programării.

Desigur definirea acestor patru direcții este insuficient de precisă și ea reprezintă subiectul a numeroase completări și dezbateri științifice. Important este faptul că ele sintetizează principalele direcții de perfecționare a tehnologiei calculatoarelor electronice, urmărirea cărora va conduce la realizarea unor calculatoare care să se deosebească esențial de predecesoarele lor pentru a putea vorbi de o nouă generație. Fără îndoială subli-



niem încă odată că una din condițiile realizării acestei noi generații este progresul în domeniul circuitelor integrate pe scară foarte largă (VLSI), cu care să se poată implementa funcțiuni de nivelul celor mai sus menționate.

În elaborarea noilor produse de tehnică de calcul, cercetătorii noștri, care urmăresc cu atenție și participă la efortul internațional de definire a noii generații, își propun să adopte soluții care să mențină nivelul tehnic al calculatoarelor românești, să aplice în practică tot ceea ce amplul efort de definire al generației a cincea va confirma ca direcție de dezvoltare a tehnicii de calcul.

*dr. ing. Vasile Baltac*



## C O N T E N T S

MIHAI DRĂGĂNESCU, Fifth-generation computer : a technological, cultural and political event 9

ADRIAN DAVIDOVICIU, An overview on the Japanese project for development of the fifth-generation computer 18

EMIL TUDOR, LUCIANNICA, MIHAI MÂRŞANU, Architectural concepts of the fifth-generation computers 47

VICTOR MEGHEŞAN, NICOLAE COSTAKE, About the 5th generation computers and the Romanian computer industry 60

VICTOR MEGHEŞAN, NICOLAE COSTAKE, MIHAI MÂRŞANU, Directions in the development of computer industry of Romania at the 1990 horizon 75

GORUN MANOLESCU, Fifth-generation computer systems and some problems of CAD systems 88

ADRIAN PETRESCU, Systolic data-processing systems 101

GHEORGHE M. ŞTEFAN, AUREL PĂUN, Function-structure merging as a mechanism of architecture evolution 113

RADU M. BÂRSAN, Large-scale integrated circuits technology 136

CRISTIAN GIUMALE, Fifth-generation computers — a step towards natural programming? 146

AUREL PĂUN, GHEORGHE ŞTEFAN, ANDY BIRNBAUM, VIRGIL BISTRICEANU, DIALISP — an experiment with a nonconventional LISP machine 160

POSTFACE 177



## СОДЕРЖАНИЕ

МИХАЙ ДРЭГЭНЕСКУ, Компьютеры пятого поколения: событие технологического, культурного и политического значения	9
АДРИАН ДАВИДОВИЧИУ, Описание японского компьютера пятого поколения	18
ЭМИЛ ТУДОР, ЛУЧИАН НИКА, МИХАЙ МЫРШАНУ, Концепции об архитектуре систем исчисления пятого поколения	47
ВИКТОР МЕГЕШАН, НИКОЛАЕ КОСТАКЕ, Компьютеры пятого поколения и румынская электронновычислительная промышленность	60
ВИКТОР МЕГЕШАН, НИКОЛАЕ КОСТАКЕ, МИХАЙ МЫРШАНУ, Направления развития румынской электронновычислительной техники на уровне 1990 г.	75
ГОРУН МАНОЛЕСКУ, Компьютеры пятого поколения и некоторые проблемы исследования ассистированного проектирования компьютера	88
АДРИАН ПЕТРЕСКУ, Системные системы обработки данных	101
ГЕОРГЕ М. ШТЕФАН, АУРЕЛ ПЭУН, Совместимость функция-структура как механизм архитектурной эволюции	113
РАДУ М. БЫРСАН, Технология для интегрированных широкомасштабных цепей	136
КРИСТИАН ДЖУМАЛЕ, Пятое поколение компьютеров — шаг к естественному программированию?	146
АУРЕЛ ПЭУН, ГЕОРГЕ М. ШТЕФАН, АНДИ БИРНБАУМ, ВИРДЖИЛ БИСТРИЧАНУ, DIALISP — опыт по нетрадиционному структурированию компьютера LISP	160
ПОСЛЕСЛОВИЕ	177